DiLaDiff: Distilled Latent-Augmented Diffusion for Language Modeling

Jean-Marie Lemercier* NVIDIA
Tomas Geffner NVIDIA
Morteza Mardani NVIDIA
Karsten Kreis NVIDIA
Arash Vahdat NVIDIA
Ante Jukić NVIDIA

Abstract

Diffusion language models intrinsically fail to capture correlations between decoded tokens, which leads to a harsh trade-off between sampling quality and throughput. To solve this issue, we propose **DiLaDiff**, a variant of masked diffusion language models with three components: (*1*) a continuous latent space with semantic capabilities, learned by an auto-encoder fine-tuned from an existing masked diffusion language model; (*2*) a latent diffusion model learning the prior over the encoder distribution; (*3*) a consistency model distilling the learned prior into a few-step latent generative model. We show that, even without distillation, our latent-guided diffusion model outperforms the masked diffusion baseline while significantly accelerating inference. Consistency distillation further lowers the computational overhead of continuous diffusion, such that the latent is generated in negligible time compared to discrete decoding.

1 Introduction

Diffusion-based generative models (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020; Song et al., 2021) have emerged as a dominant framework for modeling continuous data such as natural images (Dhariwal & Nichol, 2021), videos (Ho et al., 2022), speech (Popov et al., 2021) or protein design (Watson et al., 2023). Continuous diffusion paradigms offer many benefits, such as principled samplers through differential equation solvers as well as distillation techniques (Song et al., 2023; Geng et al., 2025b,a). Building on this success, recent works have explored extending diffusion models to categorical data (Austin et al., 2021; Campbell et al., 2022; Lou et al., 2024; Gat et al., 2024; Sahoo et al., 2024). While this formulation is well-suited to discrete data, it forgoes the aforementioned advantages of continuous latent representations. More crucially, discrete diffusion models cannot decode many tokens in parallel accurately, since the denoiser model parameterizing the posterior marginals does not capture token dependencies (Israel et al., 2025; Liu et al., 2025; Xie et al., 2026). Another line of works consider continuous diffusion on text representations, such as one-hot encodings (Lee et al., 2026), token-wise embeddings (Li et al., 2022; Gulrajani & Hashimoto, 2023; Chen et al., 2026) and contextual representations (Meshchaninov et al., 2025), thereby recovering principled sampling and distillation techniques. Despite these advantages, continuous models have been lagging behind their discrete counterparts. This has been mostly attributed to the intrinsic categorical nature of language, and also to the trade-off between expressivity and decodability of the continuous representation of text sequences (Zhou et al., 2025). On one end of the trade-off, token embeddings are easily decodable but do not extract contextual information. On the other end, contextual encodings, e.g., the output states of deep layers in a representation model (Zhou et al., 2025; Meshchaninov et al., 2025; Shariatian et al., 2025; Kang et al., 2026), provide an expressive

*Correpsonding author: jlemercier@nvidia.com

Preprint.

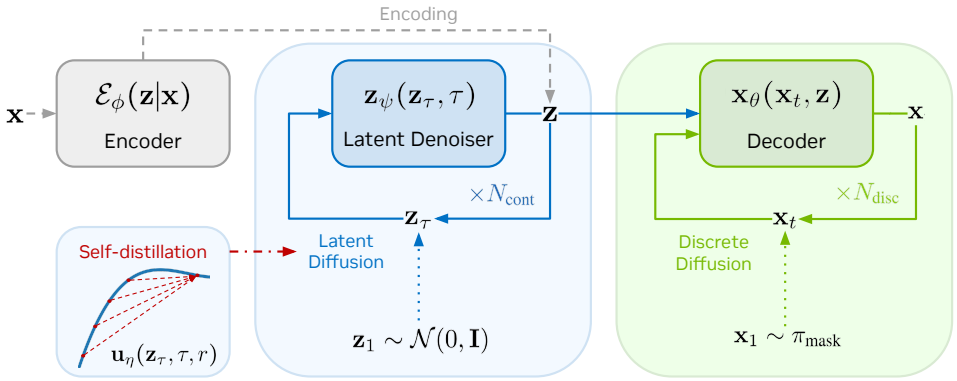

Figure 1: DiLaDiff: hybrid continuous-discrete diffusion with self-distilled latent. The latent space is crafted with encoder $\mathcal{E}_\phi$ and decoder $\mathbf{x}_\theta$ and learned a posteriori with a diffusion process with denoiser $\mathbf{z}_\psi$. The latent diffusion trajectories are further self-distilled with MeanFlow student $\mathbf{u}_\eta(\mathbf{z}_\tau, \tau, r)$.

latent but are notoriously harder to decode back to the original sequence. Zhou et al. (2025); Morris et al. (2023) suggest using a diffusion process to simplify the task of decoding the contextual latent.

Based on these insights, we propose a model combining a continuous latent prior over compressed contextual representations, and a token-space discrete decoder. The model captures global correlations via continuous diffusion over contextual representations while benefiting from robust decoding in the token space. In addition, (*1*) the decoder can be initialized from a robust pre-trained discrete diffusion model; (*2*) controllability of continuous diffusion trajectories in the latent space can be exploited; (*3*) ODE trajectories in the latent space can be distilled and continuous paradigms can be used to improve sampling. Our contributions in this work are as follows:

- We propose a recipe for training a text auto-encoder, where the decoder is initialized from an pre-trained discrete diffusion baseline. This results in a controllable and regularized latent space capturing semantic information.
- We learn the latent prior with a continuous diffusion model, yielding a hybrid continuous-discrete diffusion model suited for sampling several tokens in parallel without sacrificing text quality. The resulting model **LaDiff** (***La**tent-augmented **Diff**usion Language Model*) largely outperforms the masked diffusion baseline in unconditional generation, while accelerating inference by a factor of 7× at batch size 32.
- We leverage the properties of continuous diffusion to further distill the latent diffusion component into a few-step generative model, resulting in **DiLaDiff** (***Di**stilled **La**tent-augmented **Diff**usion Language Model)*. This is the first successful attempt at distilling ODE trajectories for contextual text latents. DiLaDiff obtains in only 5 steps a performance close to its LaDiff teacher using 200 steps. Distillation lowers the overhead of continuous diffusion, such that the latent variable is sampled in negligible time compared to discrete decoding.

2 Background

We represent language tokens as one-hot encoded vectors in $\mathcal{V} \triangleq \left\{\mathbf{v} \in \{0,1\}^K : \sum_{i=1}^K v_i = 1\right\}$ where K is the size of the vocabulary, and v_i is the i-th element of the vector $\mathbf{v}$. We use $\mathbf{x} \in \mathcal{V}^L$ to denote a sequence of L tokens and $\mathbf{x}^\ell$ is the corresponding token at the index ℓ.

2.1 Masked Diffusion Language Models

Masked diffusion language models (MDLMs) are a variant of discrete diffusion models that has emerged as a simple yet powerful paradigm for language modeling. These models learn by recovering tokens that were randomly masked in a text token sequence, using the unmasked tokens as context. MDLMs are trained using an evidence lower-bound (ELBO) on the model likelihood:

$$\mathcal{L}^\theta_{\text{MDLM}} = \mathbb{E}_{\substack{t\sim\mathcal{U}[0,1] \\ \mathbf{x}_t\sim q_t(\mathbf{x}_t|\mathbf{x})}} \frac{\dot{\alpha}_t}{1-\alpha_t} \sum_\ell \log\langle \mathbf{x}^\ell_\theta(\mathbf{x}_t), \mathbf{x}^\ell\rangle \geq -\log p_\theta(\mathbf{x})\,, \tag{1}$$

where $(\mathbf{x}_t)_{t\in[0,1]} \in \mathcal{V}^L$ is a sequence with some of the tokens in $\mathbf{x}$ replaced by a special `[MASK]` token (Sahoo et al., 2024), and $\mathbf{x}_\theta(\mathbf{x}_t)$ is the clean sequence estimate obtained from $\mathbf{x}_t$ by a denoising Transformer with parameters θ. The forward masking kernel $q_t(\cdot|\mathbf{x})$ interpolates between the clean distribution at $t = 0$ and a categorical prior $\mathbf{m}$ generating a fully masked sequence at $t = 1$. This kernel is factorized independently for each token, with the marginal distribution for the ℓ-th token:

$$q_t(\mathbf{x}_t^\ell|\mathbf{x}^\ell) = \text{Cat}\left(\mathbf{x}_t^\ell; \alpha_t\mathbf{x}^\ell + (1-\alpha_t)\mathbf{m}\right), \tag{2}$$

where $\mathbf{m}$ is a one-hot encoding of the `[MASK]` token. The proportion of masked tokens at each time step t is given by the commonly used linear noise schedule $\alpha_t = 1 - t$. Samples are obtained through the reverse posterior $q_{s|t}(\mathbf{x}_s \mid \mathbf{x}_t)$, which can be marginalized as:

$$q_{s|t}(\mathbf{x}_s \mid \mathbf{x}_t) = \int q_{s|t}(\mathbf{x}_s \mid \mathbf{x}_t, \mathbf{x}_0) q(\mathbf{x}_0|\mathbf{x}_t) \mathrm{d}\mathbf{x}_0 \,. \tag{3}$$

This posterior is intractable at inference time since we have no access to the true denoiser $q(\mathbf{x}_0|\mathbf{x}_t)$. Using a DDPM-style parameterization (Ho et al., 2020), MDLMs approximate the conditional denoiser $q(\mathbf{x}_0|\mathbf{x}_t)$ as a product of token-wise marginals, using the denoising Transformer $\mathbf{x}_\theta$:

$$q^\theta(\mathbf{x}_0|\mathbf{x}_t) \coloneqq \prod_\ell \delta\left(\mathbf{x}_0^\ell = \mathbf{x}_\theta^\ell(\mathbf{x}_t)\right), \tag{4}$$

which, in turn, yields the following approximation for the posterior:

$$q^\theta_{s|t}(\mathbf{x}_s \mid \mathbf{x}_t) \coloneqq \prod_\ell q_{s|t}\left(\mathbf{x}_s^\ell \mid \mathbf{x}_t^\ell, \mathbf{x}_0^\ell = \mathbf{x}_\theta^\ell(\mathbf{x}_t)\right), \tag{5}$$

with marginals (Sahoo et al., 2024)

$$q_{s|t}(\mathbf{x}_s^\ell|\mathbf{x}_t^\ell, \mathbf{x}_0^\ell = \mathbf{x}_\theta^\ell(\mathbf{x}_t)) \coloneqq \begin{cases} \text{Cat}(\mathbf{x}_s^\ell; \mathbf{x}_t^\ell) & \mathbf{x}_t^\ell \neq \mathbf{m}, \\ \text{Cat}\left(\mathbf{x}_s^\ell; \frac{(1-\alpha_s)\mathbf{m} + (\alpha_s - \alpha_t)\mathbf{x}_\theta^\ell(\mathbf{x}_t)}{1-\alpha_t}\right) & \mathbf{x}_t^\ell = \mathbf{m}. \end{cases} \tag{6}$$

In theory, diffusion models carry the promise of parallel generation: unlike their auto-regressive counterparts, they are not theoretically limited to accepting one token per model forward. However, in practice, diffusion models do not keep their promise because of the model mismatch between the factorized posterior $q^\theta_{s|t}$ in (5) and the true joint model $q_{s|t}$ in (3). Indeed, even if $\mathbf{x}_\theta$ perfectly recovers $\mathbf{x}_0$, the two posteriors are only equivalent if the denoiser marginals $q(\mathbf{x}_0^\ell|\mathbf{x}_t)$ are independent for all timesteps. This assumption is not true in practice, since tokens are heavily correlated in natural language, even when some clean (unmasked) context is available in $\mathbf{x}_t$. The gap between the joint posterior and its factorized approximation is especially large toward the beginning of generation where the clean context in $\mathbf{x}_t$ is scarce. This results in a trade-off between sampling quality and throughput Nie et al. (2025), as sampling more than one token per model call inevitably produces incoherent sentences.

2.2 Continuous Latent Space for Text

Another paradigm for diffusion-based language modelling is to embed the discrete text tokens into a continuous space , e.g., using auto-encoding, pre-trained contextual representations or one-hot embedding, and to perform diffusion in the latent space. Latent samples $\mathbf{z}$ are diffused using a time-dependent perturbation kernel $\tilde{q}_t(\mathbf{z}_t|\mathbf{z})$, and passed to a diffusion denoiser $\mathbf{z}_\psi$ trained with the following ELBO:

$$\mathcal{L}^\gamma_{\text{LDM}} = \mathbb{E}_{\mathbf{z}\sim\mathcal{E}(\mathbf{z}|\mathbf{x})} \mathbb{E}_{\substack{t\sim\mathcal{U}[0,1] \\ \mathbf{z}_t\sim\tilde{q}_t(\mathbf{z}_t|\mathbf{z})}} \|\mathbf{z}_\psi(\mathbf{z}_t, t) - \mathbf{z}\|_2^2 \,. \tag{7}$$

At inference, a latent $\mathbf{z}$ is obtained by reverse diffusion, then passed to the text decoder, e.g., learned decoder in the case of auto-encoding, or `argmax` operator in the case of one-hot embeddings. This formulation enjoys many properties of continuous diffusion, in particular the ability to distill the resulting probability-flow ODE trajectory (Song et al., 2023; Geng et al., 2025a).

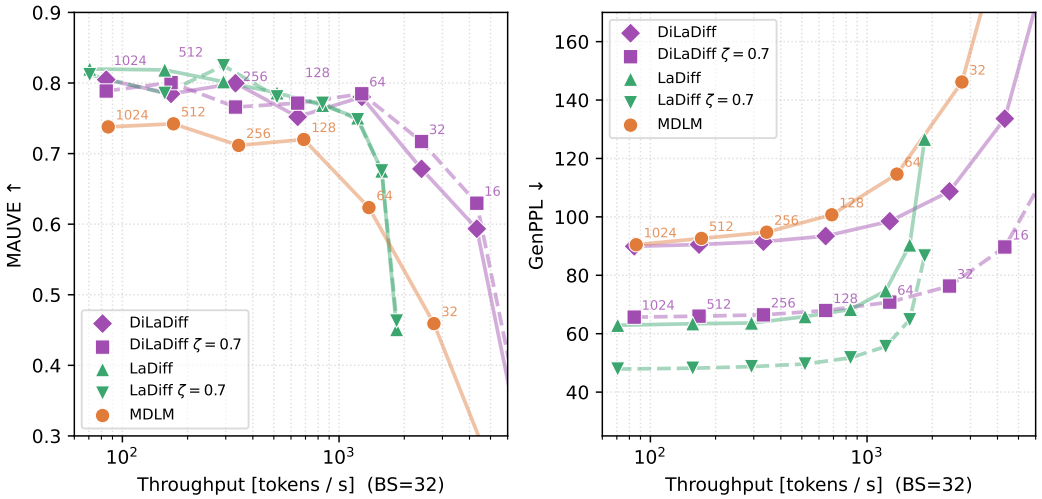

Figure 2: Speed-Quality Pareto frontier, for batch size BS = 32. Labels next to points denote the number of discrete decoding steps N_{disc}. ζ is the temperature for scaling the discrete decoder's logits.

3 Latent-Augmented Diffusion Language Models

3.1 Auto-encoding text

We propose to condition a masked diffusion language model by a latent learned via auto-encoding. The training loss emerges from combining traditional auto-encoding with (1):

$$\mathcal{L}^{\phi,\theta}_{\text{AE–MDLM}} = \mathbb{E}_{\substack{t\sim\mathcal{U}[0,1]\\ \mathbf{x}_t\sim q_t(\mathbf{x}_t|\mathbf{x})}} \frac{\dot{\alpha}_t}{1-\alpha_t} \mathbb{E}_{\mathbf{z}\sim\mathcal{E}_\phi(\mathbf{z}|\mathbf{x})} \sum_\ell \log\langle \mathbf{x}^\ell_\theta(\mathbf{x}_t,\mathbf{z}), \mathbf{x}^\ell\rangle\,. \tag{8}$$

The introduction of the latent channel $\mathbf{z}$ effectively turns the point-wise DDPM denoiser (4) into a denoiser conditioned on the latent $\mathbf{z}$:

$$q^\theta(\mathbf{x}_0|\mathbf{x}_t) := \int \delta\left(\mathbf{x}_0 = \mathbf{x}_\theta(\mathbf{x}_t,\mathbf{z})\right) p(\mathbf{z})\mathrm{d}\mathbf{z} = \int \Big(\prod_\ell \delta\left(\mathbf{x}^\ell_0 = \mathbf{x}^\ell_\theta(\mathbf{x}_t,\mathbf{z})\right)\Big) p(\mathbf{z})\mathrm{d}\mathbf{z}\,, \tag{9}$$

where $p(\mathbf{z})$ is the latent prior. This yields in turn the following posterior:

$$q^\theta_{s|t}(\mathbf{x}_s|\mathbf{x}_t) := \int \Big(\prod_\ell q_{s|t}\left(\mathbf{x}^\ell_s|\mathbf{x}^\ell_t, \mathbf{x}^\ell_0 = \mathbf{x}^\ell_\theta(\mathbf{x}_t,\mathbf{z})\right)\Big) p(\mathbf{z})\mathrm{d}\mathbf{z}\,, \tag{10}$$

Contrarily to (5), the posterior (10) truly factorizes over independent tokens *conditioned on the latent*, since the latent $\mathbf{z}$ captures the token correlations in $\mathbf{x}_0$. In particular, the token correlations captured by the latent $\mathbf{z}$ are crucial for high masking ratios t, i.e., when clean context is scarce, since this is where the mismatch between the joint denoiser $q(\mathbf{x}_0|\mathbf{x}_t)$ and its marginals is the largest (see Figure 7). The full proof is in Appendix A.

Latent space regularization We base this work on Meshchaninov et al. (2025), where pre-trained BERT hidden states are passed as input features to a Perceiver-inspired encoder (Jaegle et al., 2021) which further compresses the resulting text representation. Such representations cannot be directly learned via diffusion because of the fundamental trade-off between representation capacity and smoothness, well documented in the auto-encoding literature (Kingma & Welling, 2014). Solely optimizing for reconstruction naturally pushes latents away from each other, resulting in a latent space too sparse for a generative model to learn. We therefore employ the same robustness-enforcing heuristics as Meshchaninov et al. (2025): (*1*) coordinate-wise normalization of BERT features $\bar{\mathbf{x}}$ and latents $\mathbf{z}$; (*2*) random masking of BERT features with probability $p^{\bar{\mathbf{x}}}_{\text{mask}}$, and similarly for latents with probability $p^{\mathbf{z}}_{\text{mask}}$; (*3*) injection of Gaussian noise into BERT features, with standard deviation $\sigma^{\bar{\mathbf{x}}}_{\text{reg}}$. Finally, we further regularize training by employing a pre-trained MDLM decoder, and replacing the latent channel altogether with normally distributed noise, with chance $p^{\mathbf{z}}_{\text{dropout}}$. More details and pseudo-code for auto-encoder training can be found in Section 4 and Appendix C.1.

3.2 Modelling text latents

The clean sequence is not available at inference, and therefore we need to learn the latent prior $p(\mathbf{z})$. We propose to use the diffusion framework presented in Section 2.2 for learning text latents produced by our auto-encoder, i.e., learn a parametric prior $p_\gamma(\mathbf{z}) \approx \mathbb{E}_{\mathbf{x}}[p_\phi(\mathbf{z}|\mathbf{x})]$. As opposed to Xie et al. (2026) where $\mathbf{z}$ is directly sampled from an isotropic normal distribution, our model can capture complex dependencies using a learned latent prior, and therefore provide better guidance to the decoder. The proposed **LaDiff** model (**La**tent-augmented **Diff**usion Language Model) offers a natural interpolation scheme between pure continuous diffusion in the latent space and discrete mask-based diffusion in the token space. Pseudo-code and training details of the proposed model are listed in Appendix C.2.

At inference time, we first sample a latent sample $\mathbf{z}$ by solving the reverse probability-flow ODE using our latent denoiser $\mathbf{z}_\psi$ (trained with (7)) and time discretization $\{\tau_{0\le m\le N_{\text{cont}}}\}$, where N_{cont} is the number of continuous diffusion steps. Updates have the following form:

$$\forall m = N_{\text{cont}}, \dots, 1: \quad \mathbf{z}_{\tau_{m-1}} = \text{ODESolver}(\mathbf{z}_{\tau_m}, \tau_{m-1}, \tau_m, \mathbf{z}_\psi)\,. \tag{11}$$

We then perform ancestral sampling in the token space, starting from a fully masked sequence and progressively sampling the reverse diffusion posterior (10) with time discretization $\{t_{0\le n\le N_{\text{disc}}}\}$, where N_{disc} is the number of discrete diffusion steps:

$$\forall n = N_{\text{disc}}, \dots, 1: \quad \mathbf{x}_{t_{n-1}} \sim q^\theta_{t_{n-1}|t_n}\left(\mathbf{x}_{t_{n-1}}|\mathbf{x}_{t_n}, \mathbf{x}_0 = \mathbf{x}^\ell_\theta(\mathbf{x}_{t_n}, \mathbf{z}_{\tau_0})\right)\,. \tag{12}$$

A complete pseudo-code for LaDiff sampling procedure is available in Algorithm 3.

3.3 Distilling latent trajectories

A key advantage of using continuous latents is the ability to distill the corresponding ODE trajectories. In this work, we use MeanFlow (Geng et al., 2025a) as it naturally allows multi-step generation (as opposed to consistency models). MeanFlow tasks our student **DiLaDiff** (**Di**stilled **La**tent-augmented **Di**ffusion Language Model) with learning the *average* velocity $\mathbf{u}(\mathbf{z}_t, t, r)$, i.e., the average displacement between two points t and r on the ODE path:

$$\mathbf{u}(\mathbf{z}_t, t, r) := \frac{1}{t-r}\int_r^t \mathrm{d}\mathbf{z}_\tau\,. \tag{13}$$

In the many-step regime $r \to t$, the average velocity $\mathbf{u}(\mathbf{z}_t, t, r)$ converges to the *instantaneous* velocity $\mathbf{v}(\mathbf{z}_t, t) := \frac{\mathrm{d}\mathbf{z}_\tau}{\mathrm{d}\tau}|_{\tau=t}$, so they can be used interchangeably to sample from the model. However, in the few-step regime the two velocities are not aligned anymore, and the average velocity $\mathbf{u}$ must be used in place of the instantaneous velocity $\mathbf{v}$ in order to stay on the ODE path. MeanFlows can be trained from scratch, but we distill from an existing LaDiff teacher ψ, which yields the following objective:

$$\mathcal{L}_{\text{MeanFlow}}(\eta) = \mathbb{E}\|\mathbf{u}_\eta(\mathbf{z}_t, t, r) - \text{stopgrad}(\mathbf{u}_{\text{tgt}})\|_2^2\,, \tag{14}$$

where the target $\mathbf{u}_{\text{tgt}}$ is obtained from the instantaneous velocity and the average velocity derivatives:

$$\mathbf{u}_{\text{tgt}} = \mathbf{v}_\psi(\mathbf{z}_t, t) - (t-r)\left(\mathbf{v}_\psi(\mathbf{z}_t, t)\partial_{\mathbf{z}}\mathbf{u}_\eta + \partial_t\mathbf{u}_\eta\right)\,. \tag{15}$$

Considering first-order Euler ODE sampling for simplicity, samples are obtained by replacing the teacher's instantaneous velocity $\mathbf{v}_\gamma$ with the student's average velocity $\mathbf{u}_\eta$:

$$\forall m = N_{\text{cont}} \dots, 1: \quad \mathbf{z}_{\tau_{m-1}} = \mathbf{z}_{\tau_m} + (\tau_{m-1} - \tau_m)\,\mathbf{u}_\eta(\mathbf{z}_{\tau_m}, \tau_m, \tau_{m-1})\,. \tag{16}$$

Complete training pseudo-code for DiLaDiff is available in Algorithm 5.

4 Experiments

We perform language modelling experiments on OpenWebText (Gokaslan et al., 2019), where the last 100K documents of the dataset are used as a held-out validation set. Sentences are packed to length $L = 1024$. We use the `bert-base-uncased` tokenizer, with a vocabulary size of $K = 30522$, thus matching the setup in Meshchaninov et al. (2025).

Training Discrete diffusion baselines, MDLM (Sahoo et al., 2024) and DUO (Sahoo et al., 2025), are re-implemented and trained for 1M steps with global batch size of 512 as in Sahoo et al. (2025). We use a cosine-decay learning schedule with 1000 linear warm-up steps with a maximal learning rate of 3e-4. Auto-encoders are then fine-tuned from the MDLM checkpoint for additional 200k steps with a learning rate of 5e-5. The architecture is a modified version of Meshchaninov et al. (2025). In the encoder, the latent variable $\mathbf{z}$ is learned via cross-attention to the hidden state $\mathbf{h}$. In the decoder, we insert cross-attention layers, wrapped in zero-initialized pointwise convolutional layers, to enable the decoder's hidden state to capture the latent information. For both MDLM and the auto-encoder, we replace the ELBO weighting $\frac{\dot{\alpha}_t}{1-\alpha_t}$ by -1, as (Sahoo et al., 2026) empirically demonstrate it lowers the objective variance and therefore insures better convergence. Additional implementation details for auto-encoders can be found in Appendix C.1. LaDiff is trained for 150k steps with a learning rate of 2e-4, with the auto-encoder's weights frozen. We use the parametric tanh-logSNR variance-preserving noise schedule (Hoogeboom et al., 2023) with a grid search for the warping parameter d (see Appendix D.3). Additional implementation details for schedules can be found in Appendix C.2. DiLaDiff is self-distilled for 25k steps using frozen LaDiff teachers, with a learning rate of 5e-5 and 25% pure flow-matching loss ($t = r$). Because of the modified self-conditioning mechanism in DiLaDiff, one call of DiLaDiff requires two latent denoiser NFEs. Therefore $N_{\text{cont}} = 5$ in DiLaDiff has a computational complexity similar as $N_{\text{cont}} = 10$ for LaDiff (the exact computational overhead is given in Table 2). More details regarding self-conditioning, training procedure and hyperparameters can be found in Appendix C.3.

Evaluation We solve the probability-flow ODE with first-order Euler. For few-step sampling with DiLaDiff, we use γ-sampling (Kim et al., 2024) with $\gamma = 0.8$ (see Appendix D.4). We perform ancestral sampling for discrete diffusion, with a temperature of $\zeta = 1$ and nucleus filtering with $p = 0.9$. We report generative perplexity (GenPPL) using GPT2-Large (Radford et al., 2019) as the scoring model. The GenPPL metric may be misleading in degenerate cases, e.g., when redundant text is generated (Zheng et al., 2025). Therefore, we also report token-level entropy to complement GenPPL and to detect degenerate cases. As a semantic coherence and all-in-one metric, we also report MAUVE (Pillutla et al., 2021), using the traditional `gpt2-large` contextual embeddings.

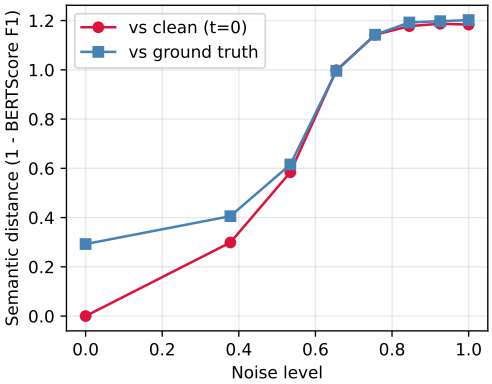

Figure 3: Semantic distance between decoded sentence of noisy versions of the same latent.

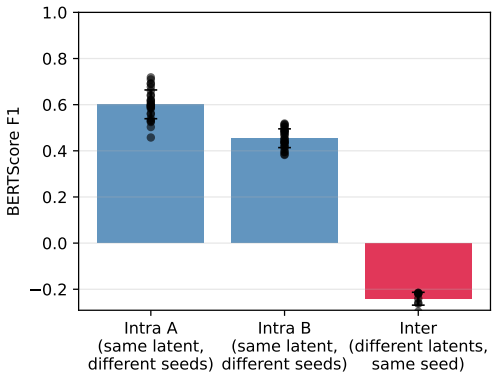

Figure 4: Semantic similarity decoded sentences from same or different latents.

4.1 Learning text representations

We evaluate the reconstruction performance of our auto-encoders on the held-out validation set of OpenWebText. The performance is evaluated in terms of the token recovery rate and the corresponding lower bound on perplexity (see Appendix G.1). As shown in Table 1, our auto-encoders have a much lower perplexity than the baseline MDLM, which is expected as they have access to the clean sequence $\mathbf{x}$. Furthermore, the perplexity and token recovery rate improve as the size of the latent space increases and reach a saturation point at around $2\times$ compression, corresponding to the latent size of $L/2 = 512$. The oracle sampling quality, evaluated by passing true encoder latents to the decoder, improves with latent space size. Similarly, using little or no regularization provides the best perplexity and oracle performance (see complete ablation in Section 4.3). However, this *does not simply translate to better generative performance* when learning the encoder prior with diffusion.

Indeed, increasing the latent channel capacity without adapting the regularization strength results in an ill-conditioned latent space which is hard to model with diffusion. Using the right regularization strength, we can reach optimal sampling quality for 2× compression and therefore use that setting for the remainder of the experiments.

Latent space analysis The auto-encoder latent is supposed to capture token correlations, i.e., some high-level representation of the sentence. We show here that this representation actually encodes some semantic information, and therefore that our encoder preserves some of the properties of the BERT feature during compression. We take a sample from the encoder's posterior distribution, corrupt it with Gaussian noise with various noise levels, then decode it with $N_{\text{disc}} = 8$ steps, keeping the decoder's stochasticity fixed. As reported on Figure 3, we observe monotonically increasing semantic distance, i.e., decreasing BERTScore-F1 (Zhang et al., 2020) between the decoded sentences (also, we qualitatively observed in this experiment that some words were being substituted for their synonyms). We also perform an analysis of the semantic diversity of our decoder when conditioned on a given latent. We sample 10 sentences from latent A with different seeds, and measure the pair-wise BERTScore-F1 within this pool of sentences. We then compare it to the pair-wise similarity between the "A" pool and a pool of sentences decoded from another latent B, using the same seed for decoding. We observe in Figure 4 that the sentences are much more closely related within a pool than between pools, but still exhibit some diversity, as shown by the deviation bars. These experiments suggest that the latent space carries semantic information and that the decoder has its own stochasticity, both of which properties are desirable for controllable and diverse generation.

4.2 Hybrid continuous-discrete diffusion

We place the performance of our hybrid models LaDiff and DiLaDiff on the speed-quality Pareto frontier in Figure 2, using a batch size BS = 32 for generation and throughput measurement. This enables taking into account the computational overhead incurred by the latent diffusion, for a fair comparison between pure discrete diffusion and our hybrid scheme. In Table 2, we compute the wall-time computational overhead of LaDiff's continuous diffusion component on a NVIDIA RTX 6000 Pro Blackwell GPU.

Batch size	LaDiff $N_{\text{cont}} = 200$ $N_{\text{disc}} = 1024$	LaDiff $N_{\text{cont}} = 200$ $N_{\text{disc}} = 64$	DiLaDiff $N_{\text{cont}} = 5$ $N_{\text{disc}} = 64$
1	0.127	1.894	0.091
8	0.075	1.175	0.053
32	0.070	1.116	0.050

Table 2: Computational overhead of latent continuous diffusion reported as a fraction of the wall-time duration of the corresponding discrete diffusion with N_{disc} steps.

In the performance-optimal regime using $N_{\text{cont}} = 200$ for continuous diffusion (i.e., no distillation) and $N_{\text{disc}} = 1024$ for discrete decoding (i.e., equal to the sequence length L) a 7% overhead is allocated to continuous diffusion in the latent space (cf. Table 2, column 2). This setting results in a 28.5 absolute (30% relative) GenPPL improvement and 0.08 absolute (10% relative) MAUVE

	Compression	PPL (↓)	%Recovery (↑)	GenPPL (↓)		MAUVE (↑)	
				Oracle	Generated	Oracle	Generated
Data	–	–	–	–	*14.8*	-	*1.00*
MDLM	–	≤ 30.0	45.2	–	90.5	-	0.74
LaDiff†	32×	≤ 6.83	53.3	61.2	120.9	0.86	0.77
LaDiff†	16×	≤ 6.83	53.3	48.7	112.5	0.89	0.80
LaDiff†	8×	≤ 3.28	61.0	45.9	80.2	0.89	0.77
LaDiff†	4×	≤ 2.58	68.3	38.9	93.9	0.89	0.74
LaDiff†	2×	≤ 2.25	**75.2**	**31.4**	85.1	**0.90**	0.73
LaDiff†	1×	≤ **2.15**	74.5	31.6	140.2	**0.90**	0.14
LaDiff	2×	≤ 3.25	69.4	41.6	**62.9**	0.87	**0.82**
LaDiff	1×	≤ 2.87	69.7	40.1	68.8	**0.90**	0.77

Table 1: Validation recovery and test performance on OpenWebText. Models marked with † have auto-encoders trained with the MildAug regularization strategy (see Section 4.3). All models use $N_{\text{cont}} = 200$ and $N_{\text{disc}} = 1024$.

improvement compared to the masked diffusion baseline (MDLM) with $N_{\text{disc}} = 1024$ steps (cf. Figure 2). When using fewer decoding steps, e.g., $N_{\text{disc}} = 64$, the proposed LaDiff profits heavily from latent guidance, and consequently outperforms the optimal setting baseline MDLM with $N_{\text{disc}} = 1024$ by 22.2 absolute (24.5% relative) GenPPL and 0.035 absolute (4% relative) MAUVE, while boasting a 7× wall-time acceleration factor. However, in this setting, the continuous diffusion using $N_{\text{cont}} = 200$ represents a 111% computational overhead compared to the discrete decoding budget using $N_{\text{disc}} = 64$.

Although latent diffusion scales favorably with batch size over discrete decoding (due to latent space compression and the absence of logits projection / categorical sampling, see Table 2), this highlights the importance of distillation. For a fixed number of $N_{\text{disc}} = 64$ discrete decoding steps, the proposed distilled model DiLaDiff obtains 0.79 MAUVE with only $N_{\text{cont}} = 5$ latent diffusion steps (resulting in $2N_{\text{cont}} = 10$ latent denoiser NFEs because of self-conditioning, see Appendix C.3), reducing the gap with its teacher LaDiff (MAUVE=0.81) with $N_{\text{cont}} = 200$. The resulting overhead of latent diffusion drops down to 5%, i.e., the latent can be computed in negligible time, while improving GenPPL by 17.3 absolute (15% relative) and MAUVE by 0.17 absolute (27% relative) compared to MDLM with the same amount of discrete decoding steps $N_{\text{disc}} = 64$.

In Table 3, we compare our methods to official baselines MDLM (Sahoo et al., 2024) and DUO (with greedy-tailed sampler) (Sahoo et al., 2025), reporting results from the corresponding papers. Furthermore, in Table 4 we also report the performance of few-step generative performance for distilled models MDLM+SDTT and DUO+DCD (Sahoo et al., 2025). It is important to note that we *do not distill the discrete decoder*, while MDLM+SDTT and DUO+DCD exclusively focus on that aspect and therefore obtain good few-decoding-steps performance. We show here that even without explicit discrete distillation, DiLaDiff is able to rival these state-of-the-art few-step methods by using latent-augmented discrete diffusion. Applying discrete distillation techniques on top of our distilled latent denoiser is a natural extension of the proposed model, and we leave it for future work.

	GenPPL (↓)	Entropy (↑)
Data	*14.8*	*5.44*
MDLM	104.5	5.63
DUO	71.7	5.22
MDLM (ours)	90.5	5.50
SDDM (ours)	75.3	5.31
LaDiff	62.9	5.40
LaDiff ζ=0.7	47.8	5.26

Table 3: Many-step generative performance on OpenWebText. $N_{\text{cont}} = 200, N_{\text{disc}} = 1024$.

	GenPPL (↓)	Entropy (↑)
Data	*14.8*	*5.44*
MDLM+SDTT	62.3	5.49
DUO+DCD	46.3	5.38
DiLaDiff	108.7	5.53
DiLaDiff ζ=0.7	76.3	5.39

Table 4: Few-step generative performance on OpenWebText. DiLaDiff using $N_{\text{cont}} = 5, N_{\text{disc}} = 32$.

Temperature sampling Using various discrete decoding strategies reveals stark contrasts between the baseline MDLM and the proposed LaDiff. As shown in Figure 5, we observe the influence of temperature logits scaling and nucleus sampling in the discrete diffusion decoder on the quality-diversity trade-off. When sampling from LaDiff, we can lower the temperature and still preserve sample entropy, which demonstrates that the latent captures diversity to a large extent (although the decoder hast its share of stochasticity, as shown in Section 4.1). In comparison, MDLM's sample entropy rapidly dwindles when decreasing the temperature, as the diversity is entirely determined by the discrete diffusion process, and nucleus sampling (with a probability threshold of 0.9) is required to obtain reasonable sample quality at temperatures approaching $\zeta = 1$.

4.3 Ablation studies

Auto-encoder training We ablate our design choices for the auto-encoding mechanisms described in Section 3.1. The influence of the regularization strategies is assessed by varying the parameters of the embedding and latent data augmentations and the results are presented in Table 5. Our best performing set of augmentation parameters is using $\{\sigma_{\text{reg}}^{\tilde{\mathbf{x}}} = 0.5, p_{\text{mask}}^{\tilde{\mathbf{x}}} = 0.7, p_{\text{mask}}^{\mathbf{z}} = 0.7\}$, as used in the proposed LaDiff model. We consider two setups with a reduced amount of augmentation. Firstly, we consider MildAug using $\{\sigma_{\text{reg}}^{\tilde{\mathbf{x}}} = 0.4, p_{\text{mask}}^{\tilde{\mathbf{x}}} = 0.5, p_{\text{mask}}^{\mathbf{z}} = 0.5\}$. Secondly, we consider SoftAug using $\{\sigma_{\text{reg}}^{\tilde{\mathbf{x}}} = 0.3, p_{\text{mask}}^{\tilde{\mathbf{x}}} = 0.3, p_{\text{mask}}^{\mathbf{z}} = 0.4\}$, as used in Meshchaninov et al. (2025)). Note that

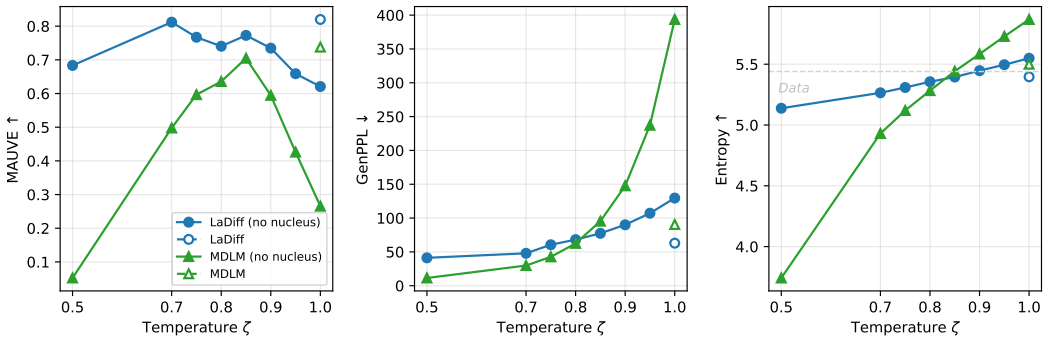

Figure 5: Temperature scaling and nucleus sampling. $N_{\text{cont}} = 200$.

			Auto-encoding		Generation	
	Ablation setup	Compression	PPL (↓)	%Recovery (↑)	GenPPL (↓)	MAUVE (↑)
LaDiff	–	2x	≤ 3.25	69.4	**62.9**	**0.82**
LaDiff	MildAug	2x	≤ 2.25	75.2	100.1	0.71
LaDiff	SoftAug	2x	≤ 1.92	81.1	98.5	0.24
LaDiff	Dropout-50%	2x	≤ **1.21**	**92.2**	101.5	0.21
LaDiff	BigEnc	2x	≤ 2.88	76.7	65.1	0.78
LaDiff	BigDec	2x	≤ 3.01	75.6	64.5	0.80

Table 5: Auto-encoder design ablations. $N_{\text{cont}} = 200$, $N_{\text{disc}} = 1024$.

BERT features $\tilde{\mathbf{x}}$ examples in a batch are *either* masked (half of the examples in a batch) or noised (the other half), not both. The strongest regularization strategy of the three yields the best results: MildAug provides reasonable generative performance, but SoftAug regularization cannot craft a latent space suitable for diffusion, yielding poor MAUVE score. Furthermore, we try reducing the chance of dropping the latent channel for decoding altogether, from 75% to 50% (cf. Dropout-50% in Table 5). This has a similar effect as using the SoftAug regularization above: token recovery is high but generative performance is much worse than base LaDiff. We also evaluated the impact of auto-encoder architecture by increasing the encoder depth from two to six layers (cf. BigEnc in Table 5) and increasing the number of cross-attention layers in the decoder from one to three (cf. BigDec in Table 5). Both these modifications do not provide significant deviations, from which we conclude than even a minimal parameterization overhead is enough to learn a qualitative latent and condition the decoder on it. For each auto-encoder, a latent denoiser is trained with an optimal, grid-searched diffusion schedule, which consistently yielded an optimal schedule parameter of $d = 10$.

5 Conclusion

We present (Di)LaDiff, a family of hybrid continuous-discrete diffusion language models. LaDiff learns a semantically-consistent latent distribution of text, and uses it to guide a token-space diffusion language model. By capturing token correlations at the sentence level, LaDiff substantially reduces the number of calls to the discrete decoder without sacrificing quality, outperforming its masked diffusion baseline while accelerating generation up to 7 times. We further push acceleration by self-distilling the latent diffusion trajectories of LaDiff, yielding the few-step generation DiLaDiff model. DiLaDiff obtains results close to the LaDiff teacher using only 5% of its latent denoiser calls, and comes within the reach of state-of-the-art models.

Limitations and future work Although DiLaDiff reduces the gap to its teacher with only a few steps, it does not match the topline teacher performance, suggesting more work is yet to be conducted on distillation. Furthermore, we do not distill the discrete decoder, which is the main reason why true few-step performance (i.e., low latent diffusion *and* low discrete decoding steps) is not state-of-the-art. This shall be explored in future work. Finally, the presented strategies for regularizing auto-encoder training (see Section 3.1), as well as the diffusion schedule design, are heuristic. We present a more principled attempt in Appendix D.3, but this requires more advanced work.

Impact statement

This paper aims to improve language modeling. Potential societal consequences include disinformation or deep fakes, but this work does not specifically address nor raise any such issue.

References

Austin, J., Johnson, D. D., Ho, J., Tarlow, D., and van den Berg, R. Structured denoising diffusion models in discrete state-spaces. *NeurIPS*, 2021.

Ben-Hamu, H., Gat, I., Severo, D., Nolte, N., and Karrer, B. Accelerated sampling from masked diffusion models via entropy bounded unmasking. *NeurIPS*, 2025.

Campbell, A., Benton, J., Bortoli, V. D., Rainforth, T., Deligiannidis, G., and Doucet, A. A continuous time framework for discrete denoising models. *NeurIPS*, 2022.

Chen, T., Zhang, R., and Hinton, G. Analog bits: Generating discrete data using diffusion models with self-conditioning. *ICLR*, 2023.

Chen, Y., Liang, C., Sui, H., Guo, R., Cheng, C., You, J., and Liu, G. Langflow: Continuous diffusion rivals discrete in language modeling, 2026. URL https://arxiv.org/abs/2604.11748.

Cheng, C., Li, J., Peng, J., and Liu, G. Categorical flow matching on statistical manifolds, 2025. URL https://arxiv.org/abs/2405.16441.

Dhariwal, P. and Nichol, A. Diffusion models beat gans on image synthesis. *NeurIPS*, 2021.

Dieleman, S., Sartran, L., Roshannai, A., Savinov, N., Ganin, Y., Richemond, P. H., Doucet, A., Strudel, R., Dyer, C., Durkan, C., Hawthorne, C., Leblond, R., Grathwohl, W., and Adler, J. Continuous diffusion for categorical data, 2022. URL https://arxiv.org/abs/2211.15089.

Gat, I., Remez, T., Shaul, N., Kreuk, F., Chen, R. T. Q., Synnaeve, G., Adi, Y., and Lipman, Y. Discrete flow matching. *NeurIPS*, 2024.

Geng, Z., Deng, M., Bai, X., Kolter, J. Z., and He, K. Mean flows for one-step generative modeling. *NeurIPS*, 2025a.

Geng, Z., Pokle, A., Luo, W., Lin, J., and Kolter, J. Z. Consistency models made easy. *ICLR*, 2025b.

Gokaslan, A., Cohen, V., Pavlick, E., and Tellex, S. Openwebtext corpus, 2019. URL http://Skylion007.github.io/OpenWebTextCorpus.

Gulrajani, I. and Hashimoto, T. B. Likelihood-based diffusion language models. *NeurIPS*, 2023.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *NeurIPS*, 2020.

Ho, J. et al. Video diffusion models. *NeurIPS*, 2022.

Hoogeboom, E., Nielsen, D., Jaini, P., Forré, P., and Welling, M. Argmax flows and multinomial diffusion: Learning categorical distributions. *NeurIPS*, 2021.

Hoogeboom, E., Heek, J., and Salimans, T. Simple diffusion: End-to-end diffusion for high resolution images. *ICML*, 2023.

Israel, D., den Broeck, G. V., and Grover, A. Accelerating diffusion llms via adaptive parallel decoding. *NeurIPS*, 2025.

Jaegle, A., Gimeno, F., Brock, A., Zisserman, A., Vinyals, O., and Carreira, J. Perceiver: General perception with iterative attention. *ICML*, 2021.

Jo, J. and Hwang, S. J. Continuous diffusion model for language modeling. *NeurIPS*, 2025.

Kang, H., Zhang, Y., Kuang, N. L., Majamaki, N., Jaitly, N., Ma, Y.-A., and Qin, L. Ladir: Latent diffusion enhances llms for text reasoning. *ICLR*, 2026.

Kim, D., Lai, C.-H., Liao, W.-H., Murata, N., Takida, Y., Uesaka, T., He, Y., Mitsufuji, Y., and Ermon, S. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. *ICLR*, 2024.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *ICLR*, 2014.

Lee, C., Yoo, J., Agarwal, M., Shah, S., Huang, J., Raghunathan, A., Hong, S., Boffi, N. M., and Kim, J. Flow map language models: One-step language modeling via continuous denoising, 2026. URL https://arxiv.org/abs/2602.16813.

Li, X. L. et al. Diffusion-lm improves controllable text generation. *NeurIPS*, 2022.

Liu, A., Broadrick, O., Niepert, M., and den Broeck, G. V. Discrete copula diffusion. *ICLR*, 2025.

Lou, A., Meng, C., and Ermon, S. Discrete diffusion modeling by estimating the ratios of the data distribution. *ICML*, 2024.

Meshchaninov, V., Chimbulatov, E., Shabalin, A., Abramov, A., and Vetrov, D. Cosmos: Compressed and smooth latent space for text diffusion modeling. *NeurIPS*, 2025.

Morris, J. X., Kuleshov, V., Shmatikov, V., and Rush, A. M. Text embeddings reveal (almost) as much as text. *EMNLP*, 2023.

Nie, S., Zhu, F., You, Z., Zhang, X., Ou, J., Hu, J., Zhou, J., Lin, Y., Wen, J.-R., and Li, C. Large language diffusion models. *NeurIPS*, 2025.

Ou, J., Nie, S., Xue, K., Zhu, F., Sun, J., Li, Z., and Li, C. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. *ICLR*, 2025.

Peebles, W. and Xie, S. Scalable diffusion models with transformers. *ICCV*, 2023.

Pillutla, K., Swayamdipta, S., Zellers, R., Thickstun, J., Welleck, S., Choi, Y., and Harchaoui, Z. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *NeurIPS*, 2021.

Popov, V., Vovk, I., Gogoryan, V., Sadekova, T., and Kudinov, M. Grad-tts: A diffusion probabilistic model for text-to-speech. *PMLR*, 2021.

Pynadath, P., Shi, J., and Zhang, R. Candi: Hybrid discrete-continuous diffusion models, 2025. URL https://arxiv.org/abs/2510.22510.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners, 2019. URL https://api.semanticscholar.org/CorpusID:160025533.

Sahoo, S. S., Arriola, M., Schiff, Y., Gokaslan, A., Marroquin, E., Chiu, J. T., Rush, A., and Kuleshov, V. Simple and effective masked diffusion language models. *NeurIPS*, 2024.

Sahoo, S. S., Deschenaux, J., Gokaslan, A., Wang, G., Chiu, J., and Kuleshov, V. The diffusion duality. *ICML*, 2025.

Sahoo, S. S., Lemercier, J.-M., Yang, Z., Deschenaux, J., Liu, J., Thickstun, J., and Jukic, A. Scaling beyond masked diffusion language models. *ICML*, 2026.

Shabalin, A., Meshchaninov, V., Chimbulatov, E., Lapikov, V., Kim, R., Bartosh, G., Molchanov, D., Markov, S., and Vetrov, D. Tencdm: Understanding the properties of the diffusion model in the space of language model encodings. *ACL*, 2025.

Shariatian, D., Durmus, A., and Peluchetti, S. Latent discrete diffusion models, 2025. URL https://arxiv.org/abs/2510.18114.

Shi, J., Han, K., Wang, Z., Doucet, A., and Titsias, M. K. Simplified and generalized masked diffusion for discrete data. *NeurIPS*, 2024.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. *ICML*, 2015.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. *NeurIPS*, 2019.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. *ICLR*, 2021.

Song, Y., Dhariwal, P., Chen, M., and Sutskever, I. Consistency models. *ICML*, 2023.

Uziel, R., Belhasin, O., Levy, I., Bercovich, A., El-Yaniv, R., Zilberstein, R., and Elad, M. Crocodil: Continuous and robust conditioned diffusion for language, 2026. URL https://arxiv.org/abs/2603.20210.

Watson, J. L., Juergens, D., Bennett, N. R., and et al. De novo design of protein structure and function with RFdiffusion. *Nature*, 620:1089–1100, 2023. doi: 10.1038/s41586-023-06415-8.

Xie, T., Xue, S., Feng, Z., Hu, T., Sun, J., Li, Z., and Zhang, C. Variational autoencoding discrete diffusion with enhanced dimensional correlations modeling. *ICLR*, 2026.

Xu, M., Geffner, T., Kreis, K., Nie, W., Xu, Y., Leskovec, J., Ermon, S., and Vahdat, A. Energy-Based Diffusion Language Models for Text Generation. *ICLR*, 2025.

Ye, J., Xie, Z., Zheng, L., Gao, J., Wu, Z., Jiang, X., Li, Z., and Kong, L. Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025.

Zhang, L., Rao, A., and Agrawala, M. Adding conditional control to text-to-image diffusion models. *ICCV*, 2023.

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. Bertscore: Evaluating text generation with bert. *ICLR*, 2020.

Zheng, H., Gong, S., Zhang, R., Chen, T., Gu, J., Zhou, M., Jaitly, N., and Zhang, Y. Continuously augmented discrete diffusion model for categorical generative modeling. *ICLR*, 2026.

Zheng, K., Chen, Y., Mao, H., Liu, M.-Y., Zhu, J., and Zhang, Q. Masked diffusion models are secretly time-agnostic masked models and exploit inaccurate categorical sampling. *ICLR*, 2025.

Zhou, C., Yang, C., Hu, Y., Wang, C., Zhang, C., Zhang, M., Mackey, L., Jaakkola, T., Bates, S., and Zhang, D. Coevolutionary continuous discrete diffusion: Make your diffusion language model a latent reasoner, 2025. URL https://arxiv.org/abs/2510.03206.

Zhou, L., Parger, M., Haque, A., and Song, J. Terminal velocity matching. *ICLR*, 2026.

Zhu, H., Chen, Z., Zhou, S., Xie, Z., Yuan, Y., Chen, S., Guo, Z., Xu, S., Zhang, H., Honavar, V., and Xiao, T. Simple denoising diffusion language models, 2026. URL https://arxiv.org/abs/2510.22926.

A Proofs

We prove here our main result in (10):

Theorem A.1 (LaDiff reverse posterior)**.** *Given a latent vector* $\mathbf{z}$ *capturing correlations in* $\mathbf{x}_0$*, and the conditional denoiser*

$$q^\theta(\mathbf{x}_0|\mathbf{x}_t,\mathbf{z}) \coloneqq \prod_\ell \delta\left(\mathbf{x}_0^\ell = \mathbf{x}_\theta^\ell(\mathbf{x}_t,\mathbf{z})\right),$$

the reverse posterior distribution of LaDiff is, for $s<t$*:*

$$q^\theta_{s|t}(\mathbf{x}_s|\mathbf{x}_t) \coloneqq \int \left(\prod_\ell q_{s|t}\left(\mathbf{x}_s^\ell|\mathbf{x}_t^\ell,\mathbf{x}_0^\ell = \mathbf{x}_\theta^\ell(\mathbf{x}_t,\mathbf{z})\right)\right)p(\mathbf{z})\mathrm{d}\mathbf{z}\,.$$

Proof. We first prove the following lemma following (Uziel et al., 2026):

Lemma A.2 (Conditional independence of reverse posterior marginals)**.** *Given a latent vector* $\mathbf{z}$ *capturing correlations in* $\mathbf{x}_0$ *through auto-encoding, such that the posterior distribution equals the product of its marginals:*

$$q^\theta(\mathbf{x}_0 \mid \mathbf{z}) = \prod_\ell q^\theta(\mathbf{x}_0^\ell \mid \mathbf{z})\,, \tag{17}$$

Then the reverse posterior conditioned on the latent $\mathbf{z}$ *factorizes into its marginals:*

$$q^\theta_{s|t}(\mathbf{x}_s \mid \mathbf{x}_t,\mathbf{z}) = \prod_\ell q^\theta_{s|t}(\mathbf{x}_s^\ell \mid \mathbf{x}_t^\ell,\mathbf{z})\,. \tag{18}$$

We derive the shape of the conditional reverse posterior distibution $q^\theta(\mathbf{x}_s|\mathbf{x}_t,\mathbf{z})$ using Bayes:

$$q^\theta(\mathbf{x}_s \mid \mathbf{x}_t,\mathbf{z}) = \frac{q(\mathbf{x}_t \mid \mathbf{x}_s,\mathbf{z})q^\theta(\mathbf{x}_s \mid \mathbf{z})}{q^\theta(\mathbf{x}_t \mid \mathbf{z})} \tag{19}$$

$$= \frac{q(\mathbf{x}_t \mid \mathbf{x}_s)q^\theta(\mathbf{x}_s \mid \mathbf{z})}{q^\theta(\mathbf{x}_t \mid \mathbf{z})}\,, \tag{20}$$

where the second equality results from $\mathbf{z}$ being an encoded view of $\mathbf{x}_0$, and from the Markov property $q(\mathbf{x}_t \mid \mathbf{x}_s,\mathbf{x}_0) = q(\mathbf{x}_t \mid \mathbf{x}_s)$.

We then derive the conditional distribution

$$q^\theta_t(\mathbf{x}_t \mid \mathbf{z}) = \int q\left(\mathbf{x}_t \mid \mathbf{x}_0,\mathbf{z}\right)q^\theta(\mathbf{x}_0 \mid \mathbf{z})\,\mathrm{d}\mathbf{x}_0 \tag{21}$$

$$= \int q\left(\mathbf{x}_t \mid \mathbf{x}_0,\mathcal{E}_\phi(\mathbf{x}_0)\right)q^\theta(\mathbf{x}_0 \mid \mathbf{z})\,\mathrm{d}\mathbf{x}_0 \tag{22}$$

$$= \int q\left(\mathbf{x}_t \mid \mathbf{x}_0\right)q^\theta(\mathbf{x}_0 \mid \mathbf{z})\,\mathrm{d}\mathbf{x}_0 \tag{23}$$

where the last equality comes from $\mathbf{z}$ being an encoded view of $\mathbf{x}_0$. We then use the key assumption that $q^\theta(\mathbf{x}_0 \mid \mathbf{z})$ factorizes into its marginals $\prod_\ell q^q(\mathbf{x}_0^\ell \mid \mathbf{z})$, which relies on the fact $\mathbf{z}$ captures the token correlations in $\mathbf{x}_0$ through auto-encoding. Using the forward kernel factorization, we obtain:

$$q^\theta(\mathbf{x}_s \mid \mathbf{z}) = \int q(\mathbf{x}_s \mid \mathbf{x}_0)q^\theta(\mathbf{x}_0 \mid \mathbf{z})\mathrm{d}\mathbf{x}_0 \tag{24}$$

$$= \int \left(\prod_\ell q(\mathbf{x}_s^\ell \mid \mathbf{x}_0)q^\theta(\mathbf{x}_0^\ell \mid \mathbf{z})\right)\mathrm{d}\mathbf{x}_0 \tag{25}$$

$$= \prod_\ell \int \left(q(\mathbf{x}_s^\ell \mid \mathbf{x}_0)q^\theta(\mathbf{x}_0^\ell \mid \mathbf{z})\mathrm{d}\mathbf{x}_0\right) \tag{26}$$

$$= \prod_\ell q^\theta(\mathbf{x}_s^\ell \mid \mathbf{z})\,, \tag{27}$$

with the fore-to-last equality resulting from Fubini's theorem. Plugging these conditional distributions in (20) and using once again the forward kernel factorization, we obtain:

$$q^\theta_{s|t}(\mathbf{x}_s \mid \mathbf{x}_t,\mathbf{z}) = \prod_\ell \frac{q(\mathbf{x}_t^\ell \mid \mathbf{x}_s^\ell)q^\theta(\mathbf{x}_s^\ell \mid \mathbf{z})}{q^\theta(\mathbf{x}_t^\ell \mid \mathbf{z})} \tag{28}$$

$$= \prod_\ell q^\theta_{s|t}(\mathbf{x}_s^\ell \mid \mathbf{x}_t^\ell,\mathbf{z})\,, \tag{29}$$

which proves Lemma A.2. In practice, that means that several tokens can be sampled in parallel and still provide consistent text since the denoiser is conditioned on an informative latent.

Parameterizing the conditional reverse marginal with our denoiser (9), we obtain

$$q^{\theta}_{s|t}(\mathbf{x}_s \mid \mathbf{x}_t, \mathbf{z}) = \prod_{\ell} q_{s|t}\big(\mathbf{x}^{\ell}_s \mid \mathbf{x}^{\ell}_t, \mathbf{x}^{\ell}_0 = \mathbf{x}^{\ell}_{\theta}(\mathbf{x}_t, \mathbf{z})\big)\,. \tag{30}$$

We emphasize that omitting the conditioning on $\mathbf{z}$ in this last equality would yield the MDLM reverse posterior in (5) with erroneous token independence assumption. The proof then follows by marginalization over $\mathbf{z}$.

□

B Additional related work

Discrete diffusion Adaptation of continuous diffusion toward direct modelling of categorical samples was first proposed in Austin et al. (2021); Hoogeboom et al. (2021), and the continuous-time Markov chain theory was developed in Campbell et al. (2022). Extensions of the framework developed rapidly, including concrete score matching (Lou et al., 2024) and discrete flow matching (Gat et al., 2024). An essential step for bridging the gap to AR models was reached with mask-based diffusion practical enhancements (Sahoo et al., 2024; Zheng et al., 2025; Shi et al., 2024; Ou et al., 2025), and with uniform-state diffusion (Sahoo et al., 2025; Zhu et al., 2026). Large mask-based diffusion language models LLaDa (Nie et al., 2025) and Dream (Ye et al., 2025) have become staples for larger-scale open research.

Continuous diffusion Efforts for embedding text diffusion in a continuous space operate on representations include probability distributions (Cheng et al., 2025; Jo & Hwang, 2025), one-hot encodings (Lee et al., 2026) , token embeddings (Li et al., 2022; Dieleman et al., 2022; Gulrajani & Hashimoto, 2023; Chen et al., 2026) and contextual representations (Meshchaninov et al., 2025; Zhou et al., 2025). Closely related to our work is Chen et al. (2026) which defines self-distilled flow maps in a continuous token-wise embedding space. The main differences are: (*1*) we consider a hybrid continuous-discrete model while they design a purely continuous model, therefore concentrating all the model complexity in the continuous modality; (2) we finetune rich contextual embeddings provided by BERT in a similar fashion as Meshchaninov et al. (2025), while they learn shallow token-wise continous embeddings. Also concurrent to our work is Lee et al. (2026), especially because they propose a self-distillation technique using flow maps. However, they operate on one-hot encodings and consider continuous-only diffusion.

Inference-time adaptations Training-free adaptation of discrete language models consists in defining a modified joint mixture distribution combining an auxiliary model (often auto-regressive) and the original diffusion marginals. Examples of these methods are discrete copula diffusion (Liu et al., 2025), energy-based diffusion (Xu et al., 2025) and adaptive parallel decoding (Israel et al., 2025).

Hybrid diffusion Hybrid models can be coarsely divided into (*1*) cross-modality diffusion, where a parallel or joint diffusion process over discrete and continuous modalities of text is conducted, and (*2*) cascaded-modality diffusion, where a continuous latent is first generated then used as guide for discrete diffusion. Cross-modality diffusion studies include Pynadath et al. (2025); Shariatian et al. (2025); Zhou et al. (2025); Zheng et al. (2026). The most related technique from this group is Zhou et al. (2025), as they use a LLM pre-trained contextual representation as their continuous modality. However, they do not provide any generation results, focusing on perplexity alone, and they do not fully generate the latent before guiding the discrete diffusion.

Our work is an example of cascaded-modality diffusion, and other such approaches are presented in Xie et al. (2026); Shariatian et al. (2025); Uziel et al. (2026). Xie et al. (2026) introduce a latent channel learned via variational auto-encoding. Although this effectively captures correlations via auto-encoding, the resulting prior is not learned a-posteriori by an additional generative model and instead a standard Gaussian sample is used as latent during inference. Furthermore, in accordance with Meshchaninov et al. (2025), we found that using traditional KL-based variational regularization for the latent space can often result in mode collapse, and is less efficient than heuristic regularization techniques. Shariatian et al. (2025) learn the latent prior with a continuous model, however they use a frozen pre-trained encoder in their language modelling experiments, train the decoder from scratch, and provide no results on complex language benchmarks such as e.g. OpenWebText. Closely related to the present work is CroCoDiL (Uziel et al., 2026) which we regard as concurrent to ours, although evolving in a different experimental framework (they focus on Python coding with 8B-scale models).

C Training details

C.1 Auto-encoding

Following Meshchaninov et al. (2025), the BERT encodings and latents are standardized coordinate-wise using statistics aggregated along training. This serves the triple-purpose of *(1)* regularizing the latent space by avoiding extreme dispersion of latent vectors; *(2)* enabling variance-preserving diffusion schedules for learning the latent and *(3)* facilitating comparison between different designs and analyses.

We use a modified version of the architecture in Meshchaninov et al. (2025). In each of its layers, the original encoder architecture models the latent variable $\mathbf{z}$ through cross-attention between the encoder's hidden state $\mathbf{h}$ and a concatenation of the hidden state $\mathbf{h}$ and the latent $\mathbf{z}$ itself, initialized as a free set of learned vectors. The original version of the decoder's architecture is symmetrical with respect to the encoder. The decoder's hidden state is initialized as the BERT embedding of a fully masked sequence, and it attends to a concatenation of itself and the latent channel through the following cross-attention mechanism:

$$\mathbf{h} \leftarrow \mathbf{h} + \mathrm{CrossAttention}(\mathbf{h}; [\mathbf{h}, \mathbf{z}])\,. \tag{31}$$

We leave the encoder unchanged but bring the following modifications to the decoder design:

- We initialize the decoder's hidden state with the partially masked sequence $\mathbf{x}_t$, in order to expose the decoder to various levels of masking, as required in our training objective (8).
- We use the pre-trained decoder's embedding table to encode the masked sequence $\mathbf{x}_t$, instead of BERT embeddings. We freeze this layer during auto-encoder training to improve stability.
- We separate the self-attention and cross-attention mechanisms in the decoder. We noticed in preliminary experiments that the original decoder architecture is not able to properly leverage the clean context in $\mathbf{x}_t$ and instead pays excessive attention to the latent $\mathbf{z}$, degrading the marginal utility of our pre-trained MDLM decoder. Therefore, we simply insert a few cross-attention layers between the original MDLM decoder's self-attention layers. As the decoder's hidden state is already modelled through self-attention throughout the decoder, we modify these cross-attention layers so as to only extract information from the latent channel. Finally, in order to avoid perturbing the decoder's hidden state early in training, we wrap these new cross-attention layers in zero-initialized pointwise convolution layers, similar as Zhang et al. (2023):

$$\mathbf{h} \leftarrow \mathbf{h} + \mathrm{ZeroConv}\left(\mathrm{CrossAttention}\left(\mathrm{ZeroConv}\left(\mathbf{h}\right); \mathbf{z}\right)\right)\,. \tag{32}$$

The encoder is warmed up for 1000 steps while the decoder is warmed up for 10000 steps, leaving enough time to kick-start the encoder before the pre-trained decoder is updated. We freeze the decoder's embedding table to further improve training stability. We finetune our auto-encoder with a batch size of 512 for 200k steps, which takes two days on 16 NVIDIA H100 GPUs.

C.2 Latent diffusion

We use the same network architecture for the latent denoiser $\mathbf{z}_\psi$ as in Shabalin et al. (2025), i.e., a diffusion Transformer (Peebles & Xie, 2023) using positional noise-level conditioning. The learning target is direct data prediction. We use self-conditioning (Chen et al., 2023) to inform the network about its past predictions during sampling: we notice that this significantly improves generation quality at virtually no extra computational cost during inference. Self-conditioning is enforced during training on 50% of batches (taken at random) by predicting the network output $\tilde{\mathbf{z}} = \mathbf{z}_\psi(\mathbf{z}_t, t, \varnothing)$ without any conditioning, detaching its gradients, and feeding this conditioning to the network for the actual prediction $\mathbf{z}_\psi(\mathbf{z}_t, t, \tilde{\mathbf{z}})$ which is used for computing the loss. Pseudo-code for LaDiff training is given in Algorithm 2.

We use variance-preserving noise schedules as the latents are standardized coordinate-wise using aggregated statistics from the auto-encoder's training. Following Meshchaninov et al. (2025), we

Algorithm 1 Auto-encoder training

Require: Frozen BERT, encoder $\mathcal{E}_\phi$, decoder $\mathbf{x}_\theta$, masking kernel $q_t(\cdot \mid \mathbf{x})$, regularization hyperparameters $p^{\tilde{\mathbf{x}}}_{\text{mask}}, \sigma^{\tilde{\mathbf{x}}}_{\text{reg}}, p^{\mathbf{z}}_{\text{mask}}, p^{\mathbf{z}}_{\text{dropout}}$, feature statistics $(\boldsymbol{\mu}_{\tilde{\mathbf{x}},\sigma_{\tilde{\mathbf{x}}}})$,

// — Encoder path —
$\tilde{\mathbf{x}} \leftarrow \text{BERT}(\mathbf{x})$ ▷ BERT's last layer representation
$\tilde{\mathbf{x}} \leftarrow (\mathbf{z} - \boldsymbol{\mu}_{\tilde{\mathbf{x}}})/\sigma_{\tilde{\mathbf{x}}}$ ▷ normalize feature
if $\text{rand}() < \frac{1}{2}$ **then**
  $\forall(\ell, k) : \tilde{x}^\ell_k \sim \delta\big(1 - \text{Bernoulli}(p^{\tilde{\mathbf{x}}}_{\text{mask}})\big)\tilde{x}^\ell_k$ ▷ feature masking
else
  $\tilde{\mathbf{x}} \sim \mathcal{N}\Big(\sqrt{1 - (\sigma^{\tilde{\mathbf{x}}}_{\text{reg}})^2}\,\tilde{\mathbf{x}};\ (\sigma^{\tilde{\mathbf{x}}}_{\text{reg}})^2\,\mathbf{I}\Big)$ ▷ feature noise
end if
$\mathbf{z} \leftarrow \mathcal{E}_\phi(\tilde{\mathbf{x}})$ ▷ encode feature
if $\text{rand}() < \frac{1}{2}$ **then**
  if $\text{rand}() < p^{\mathbf{z}}_{\text{dropout}}$ **then**
    $\mathbf{z} \leftarrow \mu_{\mathbf{z}} + \sigma_{\mathbf{z}}\boldsymbol{\eta}, \quad \boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ ▷ replace latent by Gaussian noise
  end if
else
  $\forall(\ell, k) : z^\ell_k \sim \delta\big(1 - \text{Bernoulli}(p^{\mathbf{z}}_{\text{mask}})\big)z^\ell_k$ ▷ latent masking
end if
// — Decoder path —
sample $t \sim \mathcal{U}(0, 1)$ ▷ masking ratio
sample $\mathbf{x}_t \sim q_t(\mathbf{x}_t \mid \mathbf{x})$ ▷ masked tokens
$\hat{\mathbf{x}} \leftarrow \mathbf{x}_\theta(\mathbf{x}_t, \mathbf{z})$ ▷ compute reconstruction
$\mathcal{L} \leftarrow \text{CE}(\hat{\mathbf{x}}, \mathbf{x})$ ▷ cross-entropy versus clean tokens (8)

Algorithm 2 LaDiff training

Require: encoder $\mathcal{E}_\phi$, latent denoiser $\mathbf{z}_\psi$, schedules α_t, σ_t, latent statistics $(\boldsymbol{\mu}_{\mathbf{z}}, \sigma_{\mathbf{z}})$

$\mathbf{z} \leftarrow \mathcal{E}_\phi(\mathbf{x})$ ▷ encode text
$\mathbf{z} \leftarrow (\mathbf{z} - \boldsymbol{\mu}_{\mathbf{z}})/\sigma_{\mathbf{z}}$ ▷ normalize latent
sample $t \sim \mathcal{U}(0, 1)$
sample $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
$\mathbf{z}_t \leftarrow \alpha_t\,\mathbf{z} + \sigma_t\,\boldsymbol{\epsilon}$ ▷ forward diffuse latent
if $\text{rand}() < \frac{1}{2}$ **then**
  $\tilde{\mathbf{z}} \leftarrow \mathbf{z}_\psi(\mathbf{z}_t, t, \varnothing)$ ▷ self-conditioning prediction
else
  $\tilde{\mathbf{z}} \leftarrow \varnothing$ ▷ no self-conditioning
end if
$\hat{\mathbf{z}} \leftarrow \mathbf{z}_\psi(\mathbf{z}_t, t, \tilde{\mathbf{z}})$ ▷ predict target latent
$\mathcal{L} \leftarrow \|\hat{\mathbf{z}} - \mathbf{z}\|_2^2$ ▷ MSE loss (7)

adopt the noise schedule proposed in Hoogeboom et al. (2023):

$$\text{logSNR}(t) = -d \log \tan(\pi t/2)\,, \tag{33}$$

$$\sigma^2(t) = \text{sigmoid}(-\text{logSNR}(t))\,, \tag{34}$$

$$\alpha^2(t) = \text{sigmoid}(\text{logSNR}(t))\,. \tag{35}$$

The schedule is visualized for different values of d in Figure 6, and we perform a grid search for the warping parameter d (see results in Table 6). In general, we found $d = 10$ to be optimal for most auto-encoder configurations. This means that, unlike continuous modalities, e.g., natural images or audio, most of the denoising effort should be focused on very noisy latents, which directly relates to the categorical nature of text. Indeed, even if we use contextual representations and smoothness regularization, different tokens fundamentally map to discretely distinguishable latent vectors. The amount of noise needed to allow a diffusion trajectory to jump between different latent vectors is thus typically high compared to embeddings of continuous data.

Algorithm 3 LaDiff sampling

Require: latent schedule $\{\tau_m\}_{m=0}^{M}$, discrete schedule $\{t_n\}_{n=0}^{N}$, stochasticity control $0 \le \gamma \le 1$, latent denoiser $\mathbf{z}_\psi$ discrete decoder $\mathbf{x}_\theta$.

// — Latent sampling —
Sample $\mathbf{z}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
$\hat{\mathbf{z}} \leftarrow \varnothing$ ▷ initialize self-conditioning
for $m = M, M{-}1, \ldots, 1$ **do**
  if $\gamma > 0$ **then**
    $\tau_{m-1} \leftarrow \sqrt{1-\gamma^2}\,\tau_{m-1}$ ▷ warp time grid for γ sampling
  end if
  $\hat{\mathbf{z}} \leftarrow \mathbf{z}_\psi\big(\mathbf{z}_{\tau_m}, \tau_m, \hat{\mathbf{z}}\big)$ ▷ estimate clean latent
  $\hat{\mathbf{v}} = 1/\sigma_{\tau_m}\big((\sigma_{\tau_m}\dot{\alpha}_{\tau_m} - \dot{\sigma}_{\tau_m}\alpha_{\tau_m})\hat{\mathbf{z}} + \dot{\sigma}_{\tau_m}\mathbf{z}_{\tau_m}\big)$ ▷ compute velocity
  $\mathbf{z}_{\tau_{m-1}} \leftarrow \mathbf{z}_{\tau_m} - (\tau_m - \tau_{m-1})\hat{\mathbf{v}}$ ▷ take integration step
  if $\gamma > 0$ **then**
    Sample $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    $\mathbf{z}_{\tau_{m-1}} \leftarrow \sqrt{1-\gamma^2}\,\mathbf{z}_{\tau_{m-1}} + \gamma\,\boldsymbol{\epsilon}$ ▷ re-noise
  end if
end for
// — Discrete decoding —
Sample $\mathbf{x}_{t_N} \sim \pi_{\mathbf{m}}$
$\mathbf{z} \leftarrow \sigma_{\mathbf{z}}\mathbf{z}_{\tau_0} + \boldsymbol{\mu}_{\mathbf{z}}$ ▷ denormalize latent
for $n = N, N{-}1, \ldots, 1$ **do**
  $\hat{\mathbf{x}} \leftarrow \mathbf{x}_\theta^{\ell}\big(\mathbf{x}_{t_n}, \mathbf{z}\big)$ ▷ estimate clean tokens
  $\mathbf{x}_{t_{n-1}} \sim q^{\theta}_{t_{n-1}|t_n}\big(\mathbf{x}_{t_{n-1}} \,\big|\, \mathbf{x}_{t_n}, \mathbf{x}_0 = \hat{\mathbf{x}}\big)$ ▷ unmask tokens with reverse kernel
end for
$\mathbf{x} \leftarrow \mathbf{x}_{t_0}$

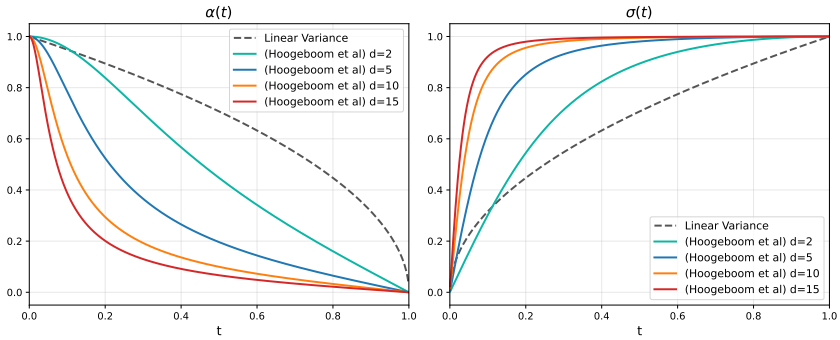

Figure 6: Diffusion schedules following the shape in Hoogeboom et al. (2023). Higher d parameters concentrate timesteps toward higher noise levels.

We train our latent diffusion model with a batch size of 512 for 500k steps, which takes two days on 8 NVIDIA H100 GPUs.

C.3 Distillation

DiLaDiff's network requires an extra time-embedding layer in order to condition the average velocity prediction $\mathbf{u}_\eta(\mathbf{z}_t, t, r)$ on terminating time step r. We initialize this layer with random weights, and embed the time difference $t - r$ instead of absolute r, following Geng et al. (2025a).

Self-conditioning is critical for performance in both LaDiff and DiLaDiff. We therefore use the *self-conditioned* LaDiff to compute the teacher target for self-distillation. This only requires an extra forward pass (with detached gradients) at training time. Furthermore, we re-use the teacher's self-conditioning layer in the student network, and pass self-conditioning as $\mathbf{z}_\eta(\mathbf{z}_t, 0) := \Phi\left(\mathbf{u}_\eta(\mathbf{z}_t, t, t), t\right)$ for 50% of training examples at random, using the property $\mathbf{u}(\mathbf{z}_t, t, t) = \mathbf{v}(\mathbf{z}_t, t)$. $\Phi(\cdot, \cdot)$ is the affine function that maps instantaneous velocities to clean data:

$$\mathbf{z} = \Phi(\mathbf{v}(\mathbf{z}_t, t), t) := \frac{\sigma_t \mathbf{v}(\mathbf{z}_t, t) - \dot{\sigma}_t \mathbf{z}_t}{\sigma_t \dot{\alpha}_t - \dot{\sigma}_t \alpha_t}, \tag{36}$$

This expression can be easily derived from the diffusion schedule

$$\mathbf{z}_t = \alpha_t \mathbf{z} + \sigma_t \boldsymbol{\epsilon} \quad \text{with} \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}) \tag{37}$$

and definition of instantaneous velocity

$$\begin{aligned}
\mathbf{v}(\mathbf{z}_t, t) &= \frac{\mathrm{d}\mathbf{z}_\tau}{\mathrm{d}\tau}\Big|_{\tau=t} && (38)\\
&= \dot{\alpha}_t \mathbf{z} + \dot{\sigma}_t \boldsymbol{\epsilon} && (39)\\
&= \dot{\alpha}_t \mathbf{z} + \dot{\sigma}_t \frac{\mathbf{z}_t - \alpha_t \mathbf{z}}{\sigma_t} && (40)\\
&= \frac{\left(\dot{\alpha}_t \sigma_t - \dot{\sigma}_t \alpha_t\right) \mathbf{z} + \dot{\sigma}_t \mathbf{z}_t}{\sigma_t}, && (41)
\end{aligned}$$

where the third equality comes from solving for $\boldsymbol{\epsilon}$ in (37). Solving for $\mathbf{z}$ in the last equality concludes the proof of (36). At sampling time (see full procedure in Algorithm 5), we dedicate an extra forward pass to compute this self-conditioning. We also experiment with more efficient designs, where we duplicate the last DiT's MLP head and train it to estimate $\mathbf{z}_\eta(\mathbf{z}_t, 0)$ or $\mathbf{u}_\eta(\mathbf{z}_t, t, 0)$ directly. However, learning this objective while simultaneously optimizing the MeanFlow loss represents an arduous task compared to the limited capacity overhead, and ultimately yielded suboptimal results (see ablations results in Appendix D.4). We dedicate more exploration of this efficient design in future work.

Training pseudo-code is given in Algorithm 4. In practice, the second term in $\mathbf{u}_{\text{tgt}}$ is efficiently computed using the true Jacobian vector product $\left(\partial_{\mathbf{z}} \mathbf{u}_\eta \; \partial_t \mathbf{u}_\eta \; \partial_r \mathbf{u}_\eta\right)^T \left(\mathbf{v}^\top \; 1 \; 0\right)$. We sample t and r in the same logit-normal distribution $\text{LogitNormal}(-1, 1)$, where samples are taken from a Gaussian $\mathcal{N}(-1, 1)$ and mapped to $[0, 1]$ using the logistic function. At random with 25% chance, r is taken equal to t, thereby reducing to classical flow matching, which improves training stability by matching the student and teacher predictions. We use loss normalization with a regularization factor of 5, effectively computing $\Delta / (\|\Delta\|_2 + 5)$ where Δ is the loss. We use a linear warmup of 10k iterations for the tangent term in the MeanFlow objective for improved stability, following Geng et al. (2025a). We self-distill a LaDiff teacher into DiLaDiff with a global batch size of 2048 for 25k steps, which takes 15 hours on 16 NVIDIA H100 GPUs.

Algorithm 4 DiLaDiff training: self-distilling LaDiff with MeanFlow

Require: Encoder $\mathcal{E}_\phi$, teacher $\mathbf{z}_\psi$, student $\mathbf{u}_\eta$, schedules α_t, σ_t, latent statistics $(\boldsymbol{\mu}_\mathbf{z}, \sigma_\mathbf{z})$

$\mathbf{z} \leftarrow \mathcal{E}_\phi(\mathbf{x})$ ▷ encode text

$\mathbf{z} \leftarrow (\mathbf{z} - \boldsymbol{\mu}_\mathbf{z})/\sigma_\mathbf{z}$ ▷ normalize latent

sample $t \sim \text{LogitNormal}(-1, 1)$

if $\text{rand}() < \frac{1}{4}$ **then**

  $r = t$ ▷ pure flow matching

else

  sample $\sim \text{LogitNormal}(-1, 1)$ and arrange timesteps s.t. $r < t$

end if

sample $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

$\mathbf{z}_t \leftarrow \alpha_t \, \mathbf{z} + \sigma_t \, \boldsymbol{\epsilon}$ ▷ forward diffuse latent

$\tilde{\mathbf{z}}_{\text{teacher}} \leftarrow \mathbf{z}_\psi(\mathbf{z}_t, t, \varnothing)$ ▷ compute teacher self-conditioning

$\mathbf{z}_{\text{teacher}} \leftarrow \mathbf{z}_\psi(\mathbf{z}_t, t, \tilde{\mathbf{z}}_{\text{teacher}})$ ▷ teacher latent prediction

$\mathbf{v}_{\text{teacher}} \leftarrow \dot{\alpha}_t \, \mathbf{z}_{\text{teacher}} + \dot{\sigma}_t \, \boldsymbol{\epsilon}$ ▷ teacher velocity (instantaneous latent velocity field)

if $\text{rand}() < \frac{1}{2}$ **then**

  $\tilde{\mathbf{z}}_{\text{student}} \leftarrow \Phi(\mathbf{u}_\eta(\mathbf{z}_t, t, r, \varnothing), t)$ ▷ update student self-conditioning

else

  $\tilde{\mathbf{z}}_{\text{student}} \leftarrow \varnothing$ ▷ disable student self-conditioning

end if

$\hat{\mathbf{u}} \leftarrow \mathbf{u}_\eta(\mathbf{z}_t, t, r, \tilde{\mathbf{z}}_{\text{student}})$ ▷ student average-velocity prediction

$\mathbf{u}_{\text{JVP}} \leftarrow \texttt{jvp}\Big(\hat{\mathbf{u}}, (\mathbf{v}_\gamma^T \; 1 \; 0)\Big)$ ▷ tangent term

$\mathbf{u}_{\text{tgt}} \leftarrow \mathbf{v}_\gamma - (t - r) \, \mathbf{u}_{\text{JVP}}$ ▷ MeanFlow target

$\mathcal{L} \leftarrow \mathbb{E}\big[\|\hat{\mathbf{u}} - \text{stopgrad}(\mathbf{u}_{\text{tgt}})\|_2^2\big]$ ▷ MeanFlow loss

Algorithm 5 DiLaDiff sampling

Require: latent schedule $\{\tau_m\}_{m=0}^{M}$, discrete schedule $\{t_n\}_{n=0}^{N}$, stochasticity control $\gamma \geq 0$, `extra_head` flag, self-conditioning threshold τ_{thr}, distilled latent model $\mathbf{u}_\eta$ discrete decoder $\mathbf{x}_\theta$.

// — Latent sampling —
Sample $\mathbf{z}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
$\tilde{\mathbf{z}} \leftarrow \varnothing$ ▷ initialize self-conditioning
for $m = M, M{-}1, \ldots, 1$ **do**
  if $\gamma > 0$ **then**
    $\tau_{m-1} \leftarrow \sqrt{1-\gamma^2}\,\tau_{m-1}$ ▷ warp time grid for γ sampling
  end if
  if `extra_head` **and** $\tau_m \geq \tau_{\text{thr}}$ **then**
    $(\hat{\mathbf{u}}, \tilde{\mathbf{z}}) \leftarrow \mathbf{u}_\eta\big(\mathbf{z}_{\tau_m}, \tau_m, \tau_{m-1}, \tilde{\mathbf{z}}\big)$ ▷ estimate average velocity + update self-conditioning
  else
    $\hat{\mathbf{u}} \leftarrow \mathbf{u}_\eta\big(\mathbf{z}_{\tau_m}, \tau_m, \tau_{m-1}, \tilde{\mathbf{z}}\big)$ ▷ only estimate average velocity
  end if
  $\mathbf{z}_{\tau_{m-1}} \leftarrow \mathbf{z}_{\tau_m} - (\tau_m - \tau_{m-1})\hat{\mathbf{u}}$ ▷ take integration step
  if not `extra_head` **or** $\tau_m < \tau_{\text{thr}}$ **then**
    $\tilde{\mathbf{z}} \leftarrow \Phi\Big(\mathbf{u}_\eta\big(\mathbf{z}_{\tau_{m-1}}, \tau_{m-1}, \tau_{m-1}, \tilde{\mathbf{z}}\big), \tau_m\Big)$ ▷ extra forward pass for self-conditioning
  end if
  if $\gamma > 0$ **then**
    Sample $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    $\mathbf{z}_{\tau_{m-1}} \leftarrow \sqrt{1-\gamma^2}\,\mathbf{z}_{\tau_{m-1}} + \gamma\,\boldsymbol{\epsilon}$ ▷ re-noise
  end if
end for
// — Discrete decoding —
Sample $\mathbf{x}_{t_N} \sim \pi_{\mathbf{m}}$
$\mathbf{z} \leftarrow \sigma_{\mathbf{z}}\mathbf{z}_{\tau_0} + \boldsymbol{\mu}_{\mathbf{z}}$ ▷ denormalize latent
for $n = N, N{-}1, \ldots, 1$ **do**
  $\hat{\mathbf{x}} \leftarrow \mathbf{x}_\theta^{\ell}\big(\mathbf{x}_{t_n}, \mathbf{z}\big)$ ▷ estimate clean tokens
  $\mathbf{x}_{t_{n-1}} \sim q_{t_{n-1}|t_n}\big(\mathbf{x}_{t_{n-1}} \,\big|\, \mathbf{x}_{t_n}, \hat{\mathbf{x}}\big)$ ▷ unmask tokens with reverse kernel
end for
$\mathbf{x} \leftarrow \mathbf{x}_{t_0}$

D Additional experiments

D.1 Dependency to masking ratio

We report in Figure 7 the reconstruction performance of our auto-encoder (with a 2× compression factor) versus pure masked diffusion, as a function of the masking ratio. Unlike MDLM, the auto-encoder is able to reconstruct the sequence reasonably well even when the masked input to the decoder has little to no unmasked context. Furthermore, we plot in Figure 8 the contributions of the cross-attention layers attending to the latent in the decoder. In the two cross-attention layers injected in the decoder (at the first and last position), the contribution of the cross-attention weights increases with the masking ratio, while the contribution of self-attention immediately preceding these cross-attention layers decreases with the masking ratio.

This demonstrates that the latent space contains most of the information needed, and therefore support our claim that the latent is particularly useful at the beginning of generation where available context is scarce.

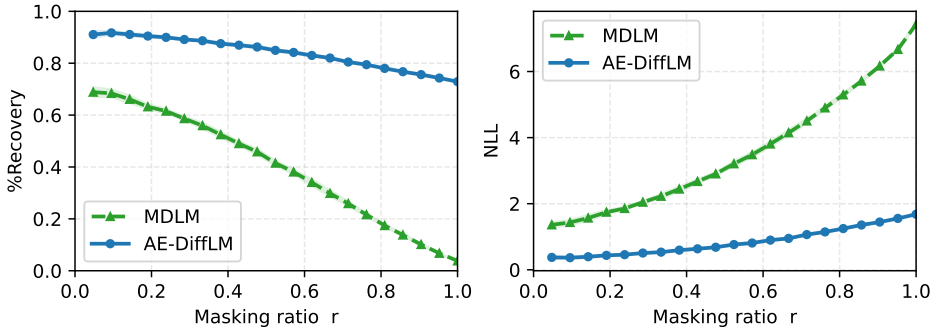

Figure 7: Performance of MDLM and auto-encoder as a function of the masking ratio. Left: Token recovery rate. Right: Negative log likelihood.

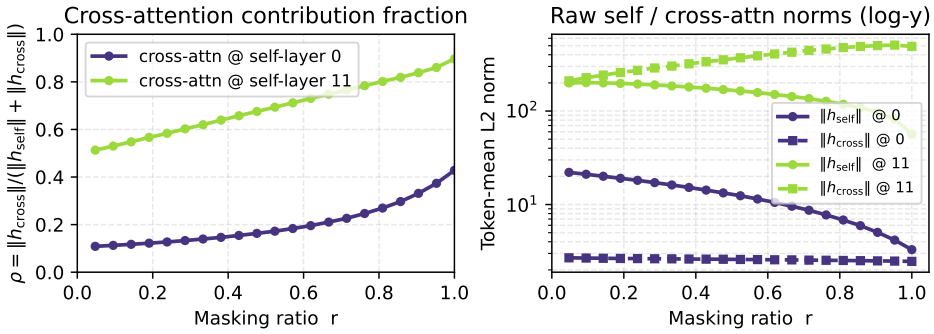

Figure 8: Cross- and self-attention contributions in decoder. Left: Fraction of the cross-attention over self-attention activations. Right: Raw norms of cross- and self-attention in first and last layer.

D.2 Confidence-based token selection

Specific to diffusion-based language modeling is the ability to use model heuristics to determine the order of token unmasking (Nie et al., 2025; Ben-Hamu et al., 2025). One common such heuristic is the model's own confidence, and it can be used to select either the top-k most confident tokens , or all tokens whose confidence exceeds a pre-determined threshold, to be decoded in the current step. We present the performance yielded by top-k confidence-based token selection with LaDiff and MDLM in Figure 9. Similar to the experiment on temperature-based logits rescaling in Section 4.1, LaDiff yields reasonable results when applying top-k confidence-based token selection, although the resulting sample entropy decreases severely from ≈5.40 down to ≈ 5.00, and MAUVE score suffers

significantly from this loss of diversity. In comparison, MDLM cannot exploit its own confidence and ends up in the trap of repeating very confident tokens such as "the", ".", etc. This yields catastrophic entropy and inexploitable results. Importantly, we dot not mean to imply that these results do simply transfer at scale: model confidence is de-facto a reliable heuristic for larger purely discrete diffusion models such as Nie et al. (2025). However, we demonstrate that, at the reference GPT2-small scale, latent guidance unlocks the ability of the discrete decoder to exploit its own confidence.

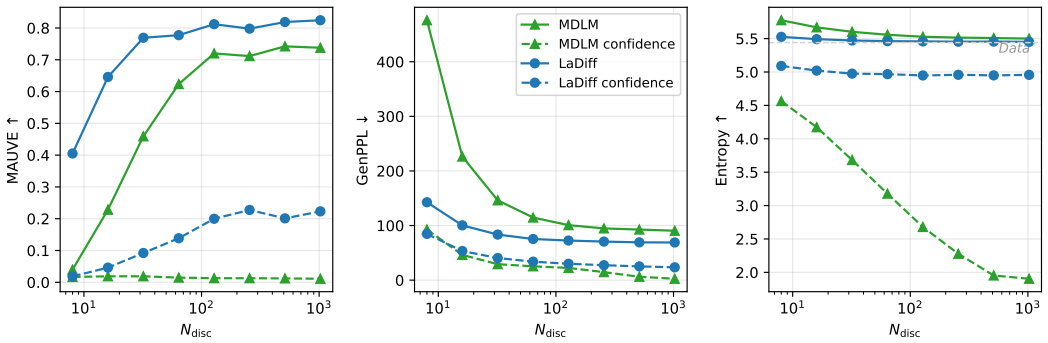

Figure 9: Confidence-based vs random token selection. $N_{\text{cont}} = 200$.

D.3 Decoder robustness and latent diffusion schedule

We visualize the robustness of our decoder to noisy text latents, for a fixed set of latent augmentations, in Figure 10. As already demonstrated in Table 1, larger latent spaces have better token recovery rates. We further show here that they exhibit similar robustness profiles, i.e. that the decoder is very robust to noise levels below a standard deviation of 0.8, above which the token recovery rate drops steeply. As the transition region is located toward $\sigma \approx 0.8$, we skew the latent diffusion schedule toward high noise levels so as not to waste training budget on very easy denoising instances. We therefore tune the d parameter of the variance-preserving schedule in (33). Results are displayed in Table 6, and as in most preliminary experiments the grid-search converged toward an optimal parameter of $d = 10$.

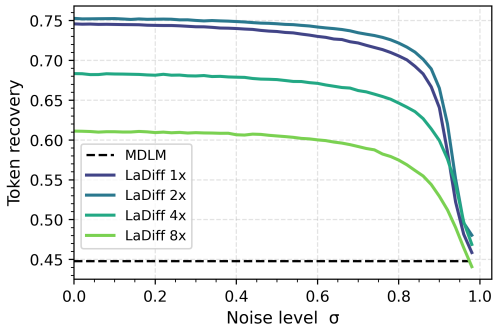

Figure 10: Decoder robustness to latent noise on OpenWebText: MDLM and LaDiff with different compression factors.

	d	GenPPL (↓)	Entropy (↑)	MAUVE (↑)
Data	-	14.8	5.44	1.00
LaDiff	2	92.0	**5.52**	0.78
LaDiff	5	71.4	5.45	0.80
LaDiff	10	**62.3**	5.40	**0.82**
LaDiff	15	75.4	5.45	0.81

Table 6: Generative performance of LaDiff when varying the noise schedule parameter d in (33). $N_{\text{cont}} = 200$, $N_{\text{disc}} = 1024$.

We propose an alternative, more principled approach, following the discussion in Lee et al. (2026), where a closed-form of the token decoding error rate is available in their one-hot encoding setup. The decoding error rate $\omega(t)$ is defined as the error rate in the token space, for a linearly-evolving noise variance $\sigma^2(t) := t$:

$$\omega(t) = 1 - \frac{\%\text{Recovery}(t)}{\%\text{Recovery}(0)}\,. \tag{42}$$

We do not have a closed-form expression of $\omega(t)$ as in Lee et al. (2026), therefore we perform a fit of the empirical recovery rate in Figure 10 to derive a diffusion time step reparameterization. We find that hyperbolic tangent functions can provide a satisfying fit of the decoding error rate function, with

only four parameters:

$$\omega_{\text{fit}}(t;\{k,t_0,\omega_{\min},\omega_{\max}\}) := \omega_{\min} + (\omega_{\max}-\omega_{\min})\frac{1+\tanh(k*(t-t_0))}{2} \tag{43}$$

which we then invert to obtain the reparameterized time:

$$\tilde{t} = \omega_{\text{fit}}^{-1}(\omega;\{k,t_0,\omega_{\min},\omega_{\max}\}) := t_0 + \frac{1}{k}\text{atanh}(2*\frac{\omega-\omega_{\min}}{\omega_{\max}-\omega_{\min}}-1)\,. \tag{44}$$

We then treat $\tilde{t}$ as our diffusion time parameterizing a linear variance-preserving schedule:

$$\sigma^2(\tilde{t}) = \tilde{t} \tag{45}$$

$$\alpha^2(\tilde{t}) = 1-\tilde{t},. \tag{46}$$

We visualize the fit in Figure 11 and the resulting time reparameterization in Figure 12.

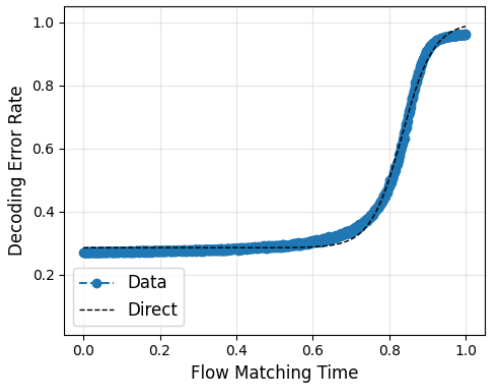

Figure 11: Empirical decoding error rate and corresponding sigmoid-tanh fit $\omega_{\text{fit}}(t)$.

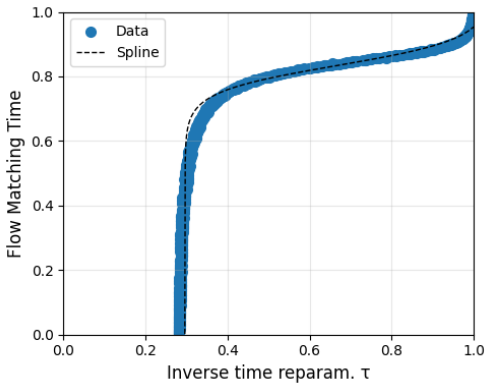

Figure 12: Time repararameterization $\tilde{t}(\omega)$

D.4 Distillation

Distillation method We tried MeanFlow (Geng et al., 2025b) as well as TVM (Zhou et al., 2026) for self-distilling LaDiff into DiLaDiff but did not notice significant differences in our experiments. We also experimented with different mean and standard deviation for the LogitNormal distribution used to sample (t,r). We ended up using MeanFlow with LogitNormal parameters $P_{\text{mean}} = -1, P_{\text{std}} = 1$. For completeness, performance is reported in Figure 13.

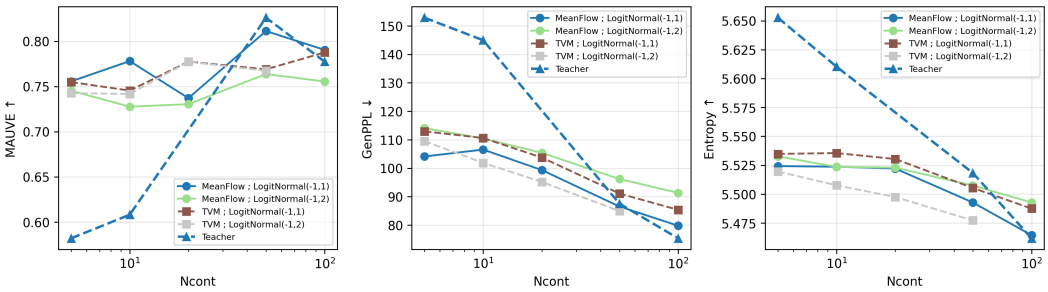

Figure 13: Ablations of self-distillation method for DiLaDiff. $N_{\text{disc}} = 64$.

γ-sampling We ablate the choice of γ for stochastic-controlled sampling (Kim et al., 2024) in our few-step DiLaDiff model. The parameter $\gamma \in [0,1]$ controls the amount of noise re-injected into the prediction:

$$\left(\tau_m \overset{\text{Denoise}}{\longrightarrow} \sqrt{1-\gamma^2}\tau_{m-1} \overset{\text{Noisify}}{\longrightarrow} \tau_{m-1}\right)_{m=1}^{M}$$

Using $\gamma = 0$ yields the deterministic ODE sampler while $\gamma = 1$ yields consistency-style stochastic sampling (at each step, jump all the way to clean data then compute the forward diffusion kernel to the next time step τ_{m-1}). Results for DiLaDiff with $N_{\text{cont}} = 5, N_{\text{disc}} = 32$ are plotted in Figure 14. Surprisingly, entropy decreases when γ increases, up to $\gamma = 0.8$, above which GenPPL explodes. $\gamma = 0.8$ seems to yield the best optimal point for MAUVE, although deviations are minimal compared to MAUVE's noise floor.

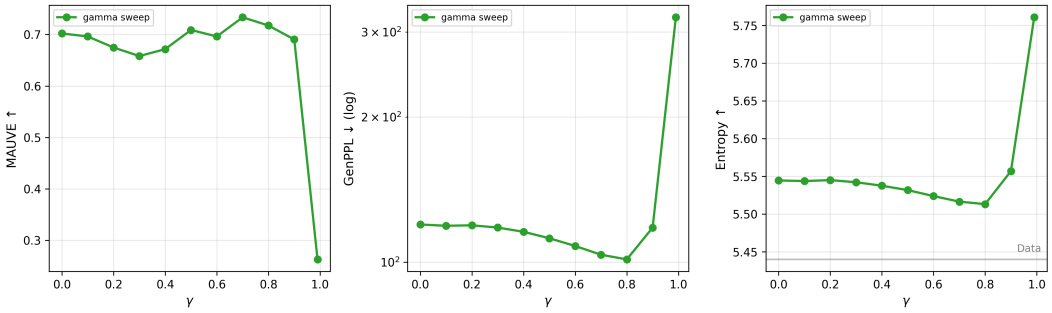

Figure 14: γ-sampling with DiLaDiff. $N_{\text{cont}} = 5, N_{\text{disc}} = 32$.

Self-conditioning We show here our experiments regarding self-conditioning design in DiLaDiff. When using an extra MLP head to estimate the self-conditioning input during training, we can either use that prediction at each step, and therefore obtain self-conditioning for free, or compute an extra forward pass to compute $\mathbf{z}_\eta(\mathbf{z}_t, 0) := \Phi\left(\mathbf{u}_\eta(\mathbf{z}_t, t, t), t\right)$, cf. Section C.3. We observed that the more sophisticated design using the extra MLP head obtained worse results, which can be due to two independent factors: (*1*) the more challenging training objective, as the student network is being asked to optimize both the MeanFlow loss and the extra regression loss $\|\mathbf{z}_\eta(\mathbf{z}_t, 0) - \mathbf{z}\|_2^2$ using the same backbone architecture; (*2*) the fundamental limitation of previous-step self-conditioning at low NFEs regime. The latter issue exists for both normal and distilled models, and comes from the mismatch between (*1*) training, where self-conditioning is computed as $\tilde{\mathbf{z}} = \mathbf{z}_\psi(\mathbf{z}_t, t, \varnothing)$ and used to refine the estimation $\mathbf{z}_\psi(\mathbf{z}_t, t, \tilde{\mathbf{z}})$ and (*2*) inference, where the self-conditioning is obtained at the current step $\tilde{\mathbf{z}} = \mathbf{z}_\psi(\mathbf{z}_{t\tau_m}, \tau_m, \tilde{\mathbf{z}}_{\text{previous}})$ and used in the *next step* to compute $\mathbf{z}_\psi\left(\mathbf{z}_{t\tau_{m-1}}, \tau_{m-1}, \tilde{\mathbf{z}}\right)$. In many-step regime, the steps are sufficiently close such that this inference-training mismatch is reasonable, however for few-step generation, this mismatch cannot be neglected. We show results for the extra-MLP approach, compared to the normal DiLaDiff, on Figure 15.

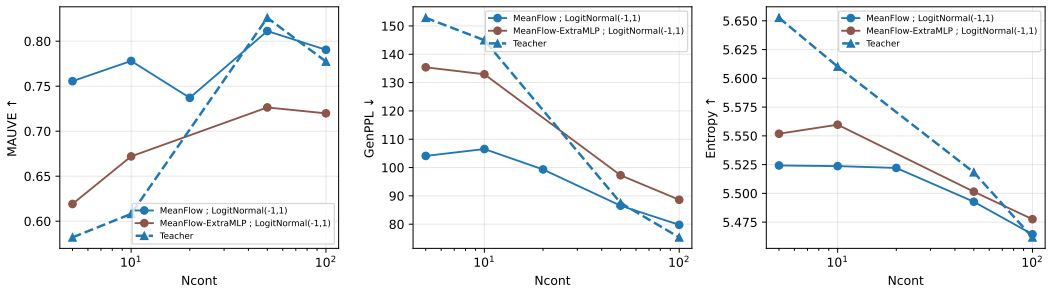

Figure 15: DiLaDiff with/without extra-MLP head for predicting self-conditioning. $N_{\text{cont}} = 5, N_{\text{disc}} = 64$.

E Baselines

For various reasons (existing and current state of implementation, choice of the BERT embedding as encoder feature, etc.), we end up using the experimental setup by Meshchaninov et al. (2025), while Sahoo et al. (2024, 2025) use the setup in Sahoo et al. (2024). For transparency, the main differences between the respective setups are listed in Table 7, and mostly result from the different DiT implementation and different tokenizer (influencing the number of embedding and logit head parameters).

	Ours	Sahoo et al. (2024)
Architecture	Absolute positional embedding QK norm	RoPe (unused) AdaLN
Tokenizer	`bert-base-uncased`	`gpt2-tokenizer`
Vocabulary size	30, 522	50, 257
Embedding / Logit parameters	22M	38M
Transformer parameters	84M	92M
Total parameters	136M	170M
Denoiser post-processing	Set mask logit to `-inf`	Set mask logit to `-inf`. Manually force logits of unmasked tokens to 0 (for corresponding logit) and `-inf` (for other logits)

Table 7: Differences between original MDLM baseline and our re-implementation.

F Pareto frontier for batch size 1

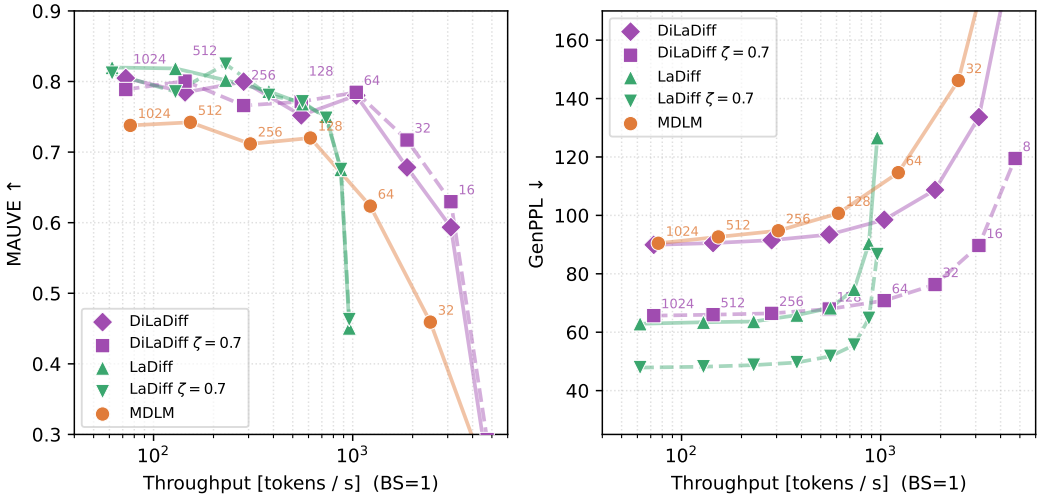

Figure 16: Speed-quality Pareto frontier for batch size BS = 1.

G Likelihood computation

G.1 Auto-encoder

We compute perplexity of our auto-encoders using the following expression:

$$\log p(\mathbf{x}) \leq \mathcal{L}^{\text{AE}}_{\text{ELBO}} := \mathcal{L}^{\text{AE}}_{\text{mELBO}} + \mathcal{L}_{\mathcal{E}} \,. \tag{47}$$

The first term is the modified ELBO corresponding to our training objective (8) with the addition of the original weighting term $\frac{\dot{\alpha}_t}{1-\alpha_t} = -\frac{1}{t}$ in equation (11) in Sahoo et al. (2024). In practice, we use the approach by Nie et al. (2025) which replaces the continuous $\frac{1}{t}, t \sim \mathcal{U}(0,1)$ with a discrete

$\frac{L}{k}, k \sim \mathcal{U}(\{1, \ldots, L\})$, thereby increasing stability:

$$\mathcal{L}^{\text{AE}}_{\text{mELBO}} = \mathbb{E}_{\substack{k \sim \mathcal{U}(\{1,\ldots,L\}) \\ \mathbf{x}_k \sim \tilde{q}_k(\mathbf{x}_k|\mathbf{x})}} \frac{L}{k} \sum_{\ell} \delta\left(\mathbf{x}_k^\ell = M\right) \log\langle \mathbf{x}_\theta^\ell(\mathbf{x}_k, \mathbf{z}), \mathbf{x}_k^\ell \rangle \,. \tag{48}$$

We slightly abuse notations to signify that the kernel $\tilde{q}_k$ masks exactly k tokens at random positions in $\mathbf{x}$, yielding $\mathbf{x}_k$. This term is evaluated using 128 Monte Carlo samples, which is enough to obtain a tight and low-variance bound according to Nie et al. (2025).

The encoder entropy vanishes because of our latent standardization procedure, when using uniform dequantization:

$$\mathcal{L}_{\mathcal{E}} := \log q_\phi^{\text{uni}}(\mathbf{z}|\mathbf{x}) = SL \log(\sigma_z) = 0 \tag{49}$$

Similarly, we compute the token recovery rate as

$$\%\text{Recovery} = \mathbb{E}_{\substack{t \sim \mathcal{U}[0,1] \\ \mathbf{x}_t \sim q_t(\mathbf{x}_t|\mathbf{x})}} \mathbb{E}_{\mathbf{z} \sim \mathcal{E}_\phi(\mathbf{z}|\mathbf{x})} \mathbb{E}_\ell \Big(\delta\left(\mathbf{x}_k^\ell = M\right) \delta\left(\mathbf{x}_\theta^\ell(\mathbf{x}_t, \mathbf{z}) = \mathbf{x}^\ell\right) \Big) \tag{50}$$

G.2 LaDiff

Although we don't present quantitative results here, we propose the following expression for computing the likelihood of LaDiff:

$$\log p(\mathbf{x}) \leq \mathcal{L}^{\text{LaDiff}}_{\text{ELBO}} := \mathcal{L}^{\text{Latent}}_{\text{PF-ODE}} + \mathcal{L}^{\text{AE}}_{\text{ELBO}} \,. \tag{51}$$

The first term is the true probability-flow ODE likelihood of our latent $\mathbf{z}$:

$$\mathcal{L}_{\text{PF-ODE}} := \log p_\theta(\mathbf{z}) = \log p_T(\mathbf{z}_T) + \int_0^T \nabla \cdot \tilde{\mathbf{z}}_\gamma(\mathbf{z}_t, t) \mathrm{d}t \,, \tag{52}$$

which is evaluated according to the protocol in Song et al. (2021). The divergence $\nabla \cdot \tilde{\mathbf{z}}_\gamma(\mathbf{z}_t, t)$ is approximated with help of Hutchinson's trace estimator (with one Rademacher-distributed noise sample) and the PF-ODE is integrated using the Runge-Kutta-45 solver.

The second term is the decoder ELBO in (48).

H Qualitative samples

LaDiff $N_{\text{cont}} = 200$, $N_{\text{disc}} = 1024$ · **GenPPL:** 62.9 · **Entropy:** 5.40

in 18 months after seeing her first deer strike about 100 kilometres from Corbrooke Avenue in Ontario. I think it's a bit cliche to think there are members of the public out there that are actually advocating against animal control. "It's just a small box, for sure, but it's just a crime. They can't keep them in themselves. They went there, it was horrible, but it was hard to think they could move it." Heather McIlgain, director of the Canadian Society of Deer and Hunters, a group that has fought against animal killings, said unusual cases are happening. She said the Canadian government has targeted some wild animals, some of which are heading to Gatehead, and the endangered animals are being being held in animal killing investigations. The association's study, combined with a collaboration with both Bell Region Public Safety and the NPP, found 456 animal sightings over the past year, mostly with government-led governments and provincial agencies that ban animal killing. "I don't know why the owner has killed an endangered animal," Gravelsome said. "There's no question about the facts. 'We are still working out which person is responsible, is this accident or non-profit,' Gravelsome said." "We've seen a lot of wild animals killed again over time. Even if they are not responsible for all animal killing investigations in the country it doesn't even seem to assess the welfare of the animals that are being examined." "The Government of Ottawa is funding assistance to all government agencies and the private sector to close the animal slaughter industry in Canada, fully shut down government offices and securely execute the rescue of farm animals." McIlgain said while she hasn't opened a door open an inquiry into how federal agencies are looking at animals, she is also asking for information about livestock and others who are planning on killing the animals. "We were just looking at farm animals. We saw a bunch of small mags, they came out of somewhere where they had stepped in," she said. "Some of the those animals had infected and turned up in traffic, and I brought up the animal, and then by laying it off, they started to move much bigger people. Then these bull lions came in, a bit bit bigger and made for thousands of clicks." She said the woman was seeing a herd of bull lions and a group of animals grazing on deer. "We pushed them several yards down the highway, so there was a bush that seemed to be circling up around." "Then we had a stump. A cule Without a stump, the cule was hard to spot," she said. Once the woman finished up the animal, the deer became injured. "It's hard to know what to see," she said. "He had a gun and I just held it, but it could easily pick one with me. Eventually the woman grabbed some of the deer and John was able to keep the animals in hand," she said. "LEASEN WORNER / FOX News Post-Morning Nurse Wayne Smith was injured when she was hit by a car in her 100-week nursing home, ABC News spoke to Cun Clhree on Monday, standing motionless at his home, where she also 'collided with her deputy dog' on Tuesday morning." "He kept mum about not telling me his his story," Clhree said. The personal caretaker told ABC News. "He's very hurt. In fact, you can't tell him how much damage he has done." Doctors are still deciding how exactly that is affected, but her apparently 50,000 year-old animal is in competent mental condition. He'll work all days, being bitten, sometimes twice a day of the week. "(Danielle) is recovering physically and mentally now and he doesn't even get to his care facility. He is sitting and I could have been an accident," said Clhree. "He's just so fragile," she said. "As soon as he feels it, he's not getting injured. He's getting hurt that much and he's back there." "Often, when people feel like someone has hit on something important, it's really not all they are." Another child, injured Monday in her 18-week-old nursing car, made the decision courtesy of Post-Morning. The child is unhappy because of her concerns, particularly the parents, who wanted to speak publicly about the issue this week. Students were to attend another school Sunday, about this week as Danielle was very good at her pet bullies and vannalism, and they also applied for custody of her mother. Topics: zoo-reality, zoo-reality, zoo, maaaa-8162, 5078-261, northern-2000, australia First Javascript

LaDiff $N_{\text{cont}} = 200, \quad N_{\text{disc}} = 64$ · **GenPPL:** 68.3 · **Entropy:** 5.40

"and I look forward to seeing that opportunity." Season resolved Schumacher, who earned his second finish in the TRT this season, as he won the race at the age of 13. "I was introduced from the start of the season and had a very bad weather. When I started training, I kicked off at Turn 6 and then dropped off a little but didn't know what else to do. I started training a little bit and was really very excited to race the next race. Now in the TRT is an amazing opportunity to continue in my four years in the RRA." "Ferrari will resume a Formula 5 TRA campaign in the new Rosso team, led by Mauricio Quesadeso (NP) who raced for 24 Spa from 2007 to 2012, winning victories in La Island and 21 Spa in the TR." "I'm very happy with the results, so I'm really looking forward to back racing after the race next year," he said of ahead of the 2017 F1 season. "It's a great time to get the start of the new season, but it brings a value to my team and I hope the will continue to continue in Le Mans." Overall, although with a bit of hard work, Ferrari returns to the form and form with France-based Örrama Ferrari FTS—TRT, $10,000 (TRAMA MEDIO). His 4th podium finish in the TRT Championship this season Total Shares $514,419 Goldman Sachs (Deutsche Bank ETF ET) SWIFT—SD) $39,599. General Counsel (NASDA), (OR) $524,614. Morgan Investments (MSFT) Goldman Sachs (AIRN) 2, 16 Chasegan Morgan Co (DHL) 12 Goldman Sachs Group (TB) 13. Morgan Stanley Capital Management (JPIF) 14 JPMorgan Chase Company (JPY) 17, 23. General Management (MSGI) 20 SwissGrann (SPDR) Wells Fargo Financial Management (FNCE) 47, 17 Goldman Sachs Accultatim Future (FINAL,) $40,000–$55,100,000. Citi Asset Management (BBBM) 1, 21 Wall Street Financial & Dividends Inc. $100,003. Morgan Stanley INK (DAK) 13. Fargo ETF (TES) 18 Morgan Stanley & Co BYC Companies (TTX) 20 JPMorgan Chase Accultati ((ADK) 13, 25. Bank of America (U.P.) 24 JP Morgan Chase Morgan Co (TB) 26 Bank of America Group (CSA) Alphabet Holding (TAG) 18, 28 Goldman Sachs Group (GK) Morgan Stanley (TK) 24 Morgan Stanley Morgan Group (TLD) 22 Morgan Stanley Global Markets (TE) 23. Morgan Stanley Banking Group (TA) 23 Standard TLDs Vanguard Goldman Morgan Group (MS) 25 Goldman Sachs Group (I) JPMorgan Morgan Co (INC) + 9, 49 JP Morgan Chase & Co (TLDF) 9, 29 The Bank of America ((BUDAQ)) Goldman Sachs & Co (NYSE: M) Then again, we have more data in data from the past results. The 23 stocks are by a third. The 23 funds are frontier-fund funds. Here are mature funds. We show how many companies have signed their companies as member companies. Goldman Sachs American Life Inc (NASDAQ) TLDs. First, we show "pioneer-pay" funds. (Jeffrey Ackman, DLOM) Morgan Stanley ETF + ((AD)) Liberty Mutual Inc (METR) Mark Morris America, President of Alpha Foundation GEE "7.5% of Morgan Stanley is the largest investment fund," as we noted in the post-Alpha US Investment Fund report. We've looked for investor funds in each index based on the most recent hedge funds. Secondly, we've only used this parameter in our initial analysis (published under SPF). Thirdly, we see "pioneer–pay" stock funds. Next, we look at those "legacy" funds. We name those funds. So here are the patterns for investors—for some reason: 1) There's 150 investors plus a top tier investor per step vs. algorithm. Our algorithm carries a fixed list price. Next, we choose confidence in trading. Since moving down, some investors will move lower. Weeks 2, 3, 5 "move down," stocks are more than 50, but by Weeks 30, investors move up. So, while we've picked the SPF over the last 4 years, we don't want investors to agree on our data and the SPF shows a good, optimistic performance. Below is the time, after Vanguard Capital Partners launched

MDLM $N_{\text{disc}} = 1024$ · **GenPPL:** 90.5 · **Entropy:** 5.50

I was investing heavily in gold due to my digital investment until recently, now many of the biggest stock bubbles investing in the world could afford to purchase. Back then we were willing to sell and to use them to buy coins. Though we did have a lot of people buying Gold coins, there are some big problems that changed people's minds. Some new crypto-trading contracts were invented back in the day there were short amounts to short selling cycles, the BO holders get much lower price than Bitcoin price. I bet most today we are very interested in coins. I assume all of cheap stuff instantly. When we invest in fiat today we got very high, but at the lowest price in any other priced piece of history. You would give yourself time to ask why gold today. Never have you be surprised if you bought gold. Have you ever invested $1 in fiat? It would be worth it if you most likely will never buy it. Today we are only in 30-something and many feel so about it. Often it counterfeit sell instantly, but value can be much higher than there ever. One of the biggest issues I discuss is the possibility of giving the money at a central bank judging the quality of the virtual currency price. I do not believe the cryptocurrency is the point to which we understand and buy makes a lot of tokens. This is true for any silver we have at all. A lot of that Commo-pricing To put it another way, a savings of $9.99 billion in annual costs per gold by making the US economy value it a lot more attractive. A major key factor in my family was earning to a BTC. The novelty of having a real gold coin was mined out of one dollar amount. This is a major change when people who rely for money on fake economic markets, it very often means fraud can get took out of the system. People buy and take virtual tokens the way they accept the money or not. They will still most likely not pay you. Now I know all about that. Dattaby, it was called, is billed as the biggest event in cycling's history. The race, known as Dattaby, will launch on 13, resulting in more than 25,000 journeys. The online stage chain Dattaby Tour Organizers has selected SportStadia, DeutschesStadia, and Hyäntra for over funding paid for taking part in the event. The special event will be expanded at the second phase and planned in a third. Next year a consortium of major airports will run on London between Channel 1 and Biao Town USA, which will open three days in two phases for spring months later. "At the end of the day, cyclists ride an atmosphere of than 2,000 miles of wild bikes and it will bring unprecedented excitement to Europe," said CEO Gerry O'Sullivan of CyclingCubb. "A national tour of Cycling is always on us. This is to be the first event in the world to take the road from London to coast back on Super France 23 for the June 21 Addero Weekend." Dattawa organisers added: "We are amazed by what all the elite cyclists will be on the three-race journey to see as the race moves towards Europe, and nothing beats a day travelling out to a bloke or the morning's coffee." "Pro cyclists from across the world will have the opportunity to take in their cycling tours with non progenitor members all over the world, and the food takeaway will become part of the thrilling food. Exhilarating accommodations along with the backbone of our cycling infrastructure, this Dattaby will put all the importance of our highly motored cycling efforts into the event. We have invited some fantastic teams together throughout our careers to compete throughout the route internationally that we are invited to compete in. We are delighted to see all cyclists available today from June, concentrating in part on keeping an authentic approach to the sport." Throughout the season there will be seven-star hotels, while hotel rooms will be used by a nighttime range of luxury, racing and fitness options. Other events happen at times in London on the same day during the season. Share this article Print Facebook Twitter Email Share this article to your email address. Show Email Click a link to the Sutton Trusty Watson Watson is demanding full access with their Travel Drivers Service. Officers have targeted people in the wake of death of a man suspected of allegedly trying to connect Uber with Uber. Dozens of arrests during Beesworth Road and Rigswell Road in Waterloo area were received today. Their own company is now not providing an investigation into their activities with the public but will be providing a drive all the way through to prepare a ticket to the event. said CEO The team is proud of our classic. By helping to organise our event, we will no doubt celebrate an elite cycling coach working in America. It

DiLaDiff $N_{\text{cont}} = 5, \quad N_{\text{disc}} = 64$ · **GenPPL:** 97.2 · **Entropy:** 5.50

nature of the comment, a speech made by Jay Leno when he split in their favour. Alan agreed in his own, of course, of winning his votes (the chart chart shows above). Nevertheless, he was able to achieve just argumentation, with no general exertation or action from class. Nonetheless, Apeball was clear that he was simply suggesting rather than implementing, to say, Wall StreetNet is attempting to force all all its members to negotiate the TLDR $1.3 trillion tax accord to GNU senators at levels far exceed the goals the party hopes voters to reach. Furthermore, it is logical that the elaborate agreement would enable the Republican to pursue a second deal through Common Core while including millions of people paying taxes, placing thousands of dollars on allegedly fraudulent schemes on their tax rolls and even as unlikely candidates for the House. At approximately the same time, the commentspunge upon the decision of the party's four chapters, who according to Wall StreetNet, declared that he disassociated, shortly after, he had lied from a pre-peace and talks about a tax deal, "looked at as a step into of various Heavens." Quite reasonable gesture for the group, although it is clear that the organization is now threatening to exit the tea party's contract with his allies, he has had no right to give up of the accounts during the 2014 elections. "All the members are already publicly acknowledging the alleged 'overspending' and beginning to pretend it is an accounting," Folder adds. The party is actually now threatening to go ahead with the plan by using spin a nebbusbitividiensis in principle as the payer in Roberto Aamer's finances. Simply put, a group of wealthy interests would acquire 5.99 millionaire shares in bonds to cover either a bill or raise its annual deficit. At that point, the on their affairs would continue from President Trumanes to the Cayman Islands, thus reducing the risk. Apparently they would have to go away if Aamer believes he is in the group with his latest proposal. The details have been denied continued negotiations. Ron Paul A. Hycock: It has been reported that the Fitz Caucus group is in contact with a giant group representing some of the voices demanding transparency in the budget: budget cuts, a border wall, and a U.S. dollar, or part of which to House President China would cost about $1.3 trillion. Media Resources: This is the view of the House Judiciary Committee on A Comprehensive Firm to De-Define the USary Sessions Report: This featured article is written by Georgia Liberty Civil Society Supreme Court Judge George C. Francis O. Lindsay. The cannabis coalition is lauded by online news outlets, covform, and local events, and constantly feeds by Christianity's support of the elections. Naturally follows the conservative Radio host's outrage over the "absurd" decision to stop bandanna Taylor Swift taking on "booing and rude shouting," the Thread in Limbaugh's host and podcast hosts, Bobby Pugh, canceled his radio show. Host Carney fired back from the offensive by complaining that the remark "offensive," a cringely demeans former rock singer Ceremonian Swift. Jordan Levivin, co-chairman of Jimmy Limmel, said Pelley had already had informal conversations with Pelley, two days after the remark, which began angering him on social media: "The goal is of a marriage equality agenda, for the party, to let people know what he she believed. This is essentially a gargantuan political push push for the failed grassroots Democratic Party on marriage—that is literally the end of one's life, rather than a legitimate overgovernment. According to Opinion and other commenters, Plante appears on his Facebook to show himself a 'champion of civility' and according to a panel of leaders, 'there is sensitivity on the price of food."' Melania Annouched against the remark, saying, "They're not talking about political unity, they're not going to say they're talking about at all." Ironically, two prominent members of Limbaugh Radio's network, Beck, stood to–face over the comment when he told Hannity that she shared the tweet publicly. "It was disgusting to hear Carney say anything like that in front of me," Beck said. "Clearly not." That apparently provoked the singer to engage his own theory of feminism, with Swift's remarks he wanted a woman to have a life, after Hannity pushed back on the offensive night. Moments after the conversation took place, pundits on the left quickly pointed to a few "conservative" chat rooms who allegedly poured gasoline on Carney's air on social, and blasted Hannity's claim they're liberal, because he supposedly opposed Swift.